2003-06

An implementation methodology and software tool for an entropy based engineering model for evolving systems

Behnke, Matthew J.

Monterey, California. Naval Postgraduate School

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

AN IMPLEMENTATION METHODOLOGY AND SOFTWARE TOOL FOR AN ENTROPY BASED ENGINEERING MODEL FOR EVOLVING SYSTEMS

by

Matthew J. Behnke

June 2003

Thesis Advisor:	Mantak Shing
Second Reader:	Christopher D. Miles

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** June 2003	**3. REPORT TYPE AND DATES COVERED** Master's Thesis	
4. TITLE AND SUBTITLE: An Implementation Methodology and Software Tool for an Entropy Based Engineering Model for Evolving Systems			**5. FUNDING NUMBERS**
6. AUTHOR(S) Behnke, Matthew J.			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000			**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A			**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.			**12b. DISTRIBUTION CODE**

13. ABSTRACT ***(maximum 200 words)***

This thesis presents a practical method for calculating and representing entropy-based metrics for a set of bibliographic records evolving over time, in support of Dr. Michael Saboe's dissertation research which addressed the ability to measure software technology transfer. The implementation of the analysis methodology for determining the information-temperature of evolving datasets containing bibliographic records is described. The information-temperature metric is based on information entropy and is used to relate the maximum complexity of a system to the current complexity.

The implementation of the analysis methodology required using data mining techniques to prepare the datasets. Additionally, since the information-temperature metric derived from Saboe's work was a new emerging concept, the data analysis methodology had to be refined several times in order to obtain the desired results. An iterative software development paradigm was used to write the application in 3 iterations using Visual Basic.

At the end of the implementation the data analysis process became systemized allowing the outlining of the steps to compute the temperature of datasets, and it is estimated that the learning curve of the analysis can be reduced by 50 percent through integration and packing of the analysis functions into a stand-alone application with an intuitive user interface.

14. SUBJECT TERMS Software Engineering, Entropy, Information Theory, Software Methodology, Data Analysis, Data Mining, Bibliographic, Tech-OASIS, Vantagepoint, Technology Transfer, Iterative Development, Information Temperature, DataThermometer, Saboe Degrees			**15. NUMBER OF PAGES** 342
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UL

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited.

AN IMPLEMENTATION METHODOLOGY AND SOFTWARE TOOL FOR AN ENTROPY BASED ENGINEERING MODEL FOR EVOLVING SYSTEMS

Matthew J. Behnke
Civilian, United States Army
B.S., Kettering University, 2001

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN SOFTWARE ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
June 2003

Author:

Matthew J. Behnke

Approved by:

Mantak Shing
Thesis Advisor

Christopher D. Miles
Second Reader

Peter Denning
Chairman, Computer Science Department

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

This thesis presents a practical method for calculating and representing entropy-based metrics for a set of bibliographic records evolving over time, in support of Dr. Michael Saboe's dissertation research which addressed the ability to measure software technology transfer. The implementation of the analysis methodology for determining the information-temperature of evolving datasets containing bibliographic records is described. The information-temperature metric is based on information entropy and is used to relate the maximum complexity of a system to the current complexity.

The implementation of the analysis methodology required using data mining techniques to prepare the datasets. Additionally, since the information-temperature metric derived from Saboe's work was a new emerging concept, the data analysis methodology had to be refined several times in order to obtain the desired results. An iterative software development paradigm was used to write the application in 3 iterations using Visual Basic.

At the end of the implementation the data analysis process became systemized allowing the outlining of the steps to compute the temperature of datasets, and it is estimated that the learning curve of the analysis can be reduced by 50 percent through integration and packing of the analysis functions into a stand-alone application with an intuitive user interface.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

Appreciation goes to all those who contributed their views and comments on the content of this thesis.

Special personal thanks to the late Michael S. Saboe, Ph. D. Without his guidance, foresight, and work this thesis would not have been possible. He will always be remembered.

Thanks to my advisors Dr. Mantak Shing and Christopher Miles for their guidance on this thesis.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. BACKGROUND

The information in this thesis includes an explanation of the problem the author worked on, the methods in which the author used to research and alleviate the problem, and the results achieved at the conclusion of the research.

This thesis presents a practical method for calculating and representing entropy-based metrics for a set of bibliographic records evolving over time in support of Dr. Michael Saboe's dissertation research which addressed the ability to measure software technology transfer. Saboe's goal was to create mechanisms utilizing information theory, communication theory, chaos control theory, and learning curve principles to determine and predict the evolution of technology. The term entropy comes from the field of information theory and appears throughout this thesis. Entropy in this context is described below in the following excerpt from Saboe's dissertation:

> Entropy as a concept can readily be seen as logical entropy (think of it as a measure of uncertainty, noise, non-signal, process inefficiencies, the percentage of work resulting in defects and requiring rework, etc) and physical or thermodynamic entropy (i.e. mixed-up-ed-ness, disorder, disorganization, etc), which is the quantity of energy not available to do work. Logical entropy is Shannon's entropy (S_H) as defined by Shannon on his treatise on communication theory (Shannon 1948). Shannon's theory says that the entropy of an information source measures how well its behavior (e.g. the next symbol in a sequence it produced) can be predicted.
>
> The definition of entropy here is related to the definition of entropy in thermodynamics. What follows is a basic review of entropy in information theory after Shannon (1948). Let X be a discrete random variable with alphabet Ξ and a probability mass function $p(x)=\Pr\{X=x\}$, $x \in \Xi$. $p(x)$ and $p(y)$ refer to two different random variables and are in fact two different probability mass functions $p_x(x)$ and $p_y(y)$.

The definition of information entropy is:

$$S_H(X) = -\sum_{x \in \Xi} p(x) \log_2 p(x)$$

> S_H is the entropy measured in bits, and the log is base 2. Log_2 will be assumed throughout unless otherwise noted. For example, the entropy of a fair coin toss is 1 bit. The convention of $0^+ \log 0^+ \to 0$ is used, which comes from continuity since $x \log x \to 0^+$, as $x \to 0^+$. The base of the log is two for the natural units of information entropy as developed by Shannon (Shannon 1948). The entropy is a function of the distribution of X. It does not depend on the actual values taken by the random variable X, but only on the probabilities.

As Dr. Michael Saboe reached the goal of his dissertation he developed a unit of measure dubbed "Saboe degrees". A Saboe degree is an information-temperature measurement that was shown to be the controlling parameter of an evolving system. Saboe demonstrated that this information-temperature is derived from a system of equations which includes both an abstract representation of a conserved property (information in terms of primitive messages) and entropy (in information units of bits). This was demonstrated through the results of applying several calculations and transformations to datasets containing bibliographic records. Saboe describes the significance of the temperature metric in the following passage from his dissertation:

> Temperature is significant because it relates the maximum complexity of a system to the current complexity. This is a proven metric that can be applied in many places to software engineering, e.g. software complexity. A direct relationship can be easily made to Halstead's metric which is familiar to software engineers. This in turn has been related to the rate humans are capable of making decisions between two choices, e.g. alphabet sets of sets of operator and operands, operators and edges, operators and flows.

The information-temperature metric is also used to describe technology transfer. The Pressure Law states that at constant volume, pressure is directly proportional to temperature meaning as pressure increases, temperature increases. Saboe defined pressure as the number of <messages> processed per node, where the <messages> represents the average output in a time step per node. A node is considered a publishing organization in the context of data presented in this thesis. Thus the temperature increases as the amount of published information increases. Theories exist stating that technology evolution can be determined by the rate and the amount of published information over

time. These theories are out of the scope of this thesis, but a list of theories noted by Dr. Saboe is found in Appendix Q. With the relationship between temperature and pressure established a higher temperature means a technology is evolving quicker than one with a lower temperature.

Methods from the field of data mining were used to obtain and prepare the bibliographic datasets, and Saboe's methodology and temperature metric were applied to the data by the author of this thesis to validate Saboe's theory. Additionally, this thesis centers on the development of a software application to solve the following problem:

A solution needs to exist that will preserve the calculation and data processing methods that produced the data analysis results shown in Dr. Michael Saboe's doctoral dissertation. This solution must increase the efficiency of the current data analysis process by minimizing the amount of user effort required to complete the analysis tasks.

B. RESEARCH QUESTIONS

This thesis focuses on providing answers to the following questions:

1. Can the data analysis process be systemized?
2. What are the steps required to determine the temperature of a set of bibliographic records?
3. Can the learning curve of the process be reduced?

C. SCOPE, LIMITATIONS, AND ASSUMPTIONS

The main outputs of this thesis are the requirements, design and architecture for the implementation of a software application named DataThermometer. DataThermometer is a data mining analysis tool that determines a bibliographic dataset's temperature. The actual implementation of a ready-for-release version will not be created for this thesis due to time and man-power constraints.

D. METHODOLOGY

The author is an employee in the Next Generation Software Technology Area (NextGen), an organization within the US Army's Tank Automotive and Armaments

Command (TACOM), who worked with Dr. Michael Saboe while he was the Associate Director of NextGen. The work involved reducing and analyzing data to be used in Saboe's doctoral dissertation. With that, the methods the author used in attacking this project included consulting experts in the data mining field, interviewing co-workers, reviewing literature concerning data mining and software development, and attending Software Engineering courses at the Naval Postgraduate School. All the research and work performed has been done within the last 18 months.

Two major challenges were encountered and overcome during the completion of this thesis. First, the information-temperature metric derived from Saboe's work was a new emerging concept. Being a new concept, the data analysis methodology had to be refined several times in order to obtain the desired results. The author chose to implement an iterative software development paradigm to facilitate the development of the evolving data analysis methods. Second, the large volume of data to be analyzed required the use of data mining techniques to prepare it for processing.

E. ORGANIZATION

The following chapters describe major specific steps of the procedure used in solving the problem. First, Chapter II presents an overview of data mining and its applicability to this thesis. Secondly, Chapter III explains the software development process utilized during the development of DataThermometer. Finally, the analysis and conclusions are presented in Chapter IV.

II. DATA MINING

The data had to be prepared before any type of calculation algorithms could be applied. The raw data had to be transformed into a structured representation. The transformation involved gathering the raw data from online databases containing a collection of scholarly references. After obtaining the raw data, the record structure of the dataset was analyzed. From determining the structure of the raw data, the author was able to create a regular expression import filter to process the records. Next, the data was fed through the filter where the data was processed and organized into a structured representation. Once the data was in this structured form, several activities followed to finish preparing the data. The activities come from the field of data mining. The next sections give a brief overview of data mining and are followed by a discussion of the data preparation activities used for this thesis.

A. WHAT IS DATA MINING

"Data Mining" is a term referring to a recently-born discipline of study. Being such a young term there is no agreed upon definition. The definition presented below is at a high enough level to effectively describe data mining.

Data mining is the process of discovering meaningful new correlations, patterns and trends by sifting through large amount of data stored in repositories, using pattern recognition technologies as well as statistical and mathematical techniques (Gartner Group). [1]

The term data mining can also be used to describe a variety of data processing tools and strategies that increase the utility of data stored in databases. While there is currently no universally accepted definition, the term is broadly used whenever a process or person attempts to discover knowledge buried within a database.

B. DATA MINING EVOLUTION

Data mining has evolved over the past 40 years from the process of analyzing data. Figure 1, below, shows the evolution began with in the 1960's with data collection and continues today with data mining.

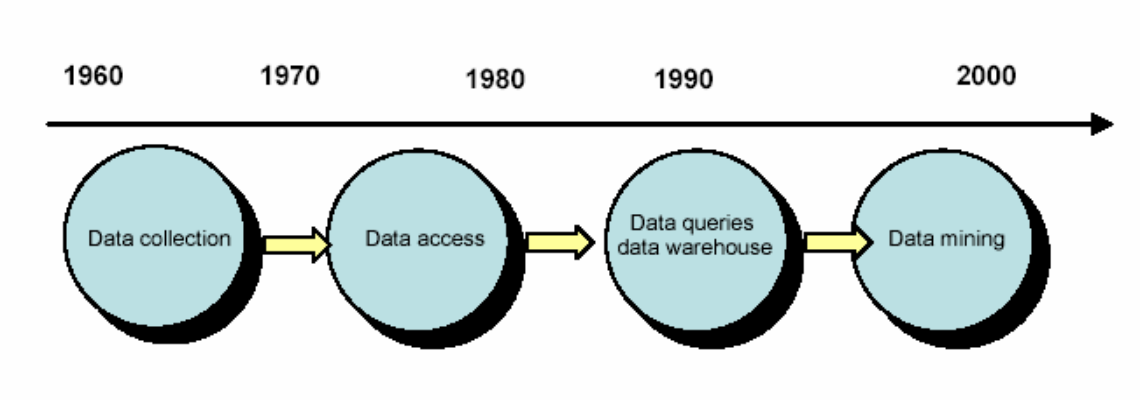


Figure 1. Evolution of data mining. [From 2]

1. Data Collection

In the 1960s development programs focused on creating applications to present standardized reports of information contained in databases. The applications retrieved and manipulated the raw data producing simple summaries to meet specific decision-making needs.

2. Data Access

During 1980s, the desire for frequent and individualized information requests brought upon queries (informational requests). Queries accessed databases in order to obtain answers to a specific question. The queries of this era were built into the database system by the system developers. This produced a substantial time gap between implementing the query and obtaining the answer from it. Many database old-timers can recall the interminable information delays from the lack of ad-hoc queries (see Data Queries below). [3]

3. Data Queries

Later, in the 1990s, the need for quick answers to spur-of-the-moment questions (ad hoc queries) arose. [3] Users wanted information to be "just-in-time" to correlate with their production and decision-making processes. That meant that not all of the users' informational needs could be preprogrammed into the system. At this stage, users began to write their own queries to extract the information that they needed from the database.

4. Data Mining

In the last few years, needs changed once again. Companies realized that they had accumulated volumes of information; and, as a result, the search began for tools and techniques that automatically identify and find relationships within these huge volumes of data. The techniques took the responsibility of finding data relationships out of the user's hands and left the decisions up to the software. [4] The removal of this tedious task from the user, allows users to focus their attention on other areas such as interpreting what the found relationships mean.

Data mining tools were first developed to help scientists find meaningful relationships or patterns from huge amounts of data that, if done in a traditional way, would require much time and many resources to find.

Data mining, in many ways, is fundamentally the adaptation of machine learning techniques to business applications. Data mining is best described as the union of historical and recent developments in statistics, AI, and machine learning. These techniques are then used together to study data and find previously-hidden trends or patterns within. Data mining is finding increasing acceptance in science and business areas which need to analyze large amounts of data to discover trends which they could not otherwise find.

Table 1, below, outlines questions of each era, discussed above, along with enabling technologies used to find answers to those questions.

Evolutionary Step	Business Question	Enabling Technologies	Product Providers	Characteristics
Data Collection (1960s)	"What was my total revenue in the last five years?"	Computers, tapes, disks	IBM, CDC	Retrospective, static data delivery
Data Access (1980s)	"What were unit sales in New England last March?"	Relational databases (RDBMS), Structured Query Language (SQL), ODBC	Oracle, Sybase, Informix, IBM, Microsoft	Retrospective, dynamic data delivery at record level
Data Warehousing &Decision Support (1990s)	"What were unit sales in New England last March? Drill down to Boston."	On-line analytic processing (OLAP), multidimensional databases, data warehouses	Pilot, Comshare, Arbor, Cognos, Microstrategy	Retrospective, dynamic data delivery at multiple levels
Data Mining (Emerging Today)	"What's likely to happen to Boston unit sales next month? Why?"	Advanced algorithms, multiprocessor computers, massive databases	Pilot, Lockheed, IBM, SGI, numerous startups (nascent industry)	Prospective, proactive information delivery

Table 1. Evolution of data mining techniques. [From 5]

C. DATA MINING ROOTS

As mentioned in the previous section, data mining is a multidisciplinary field at the intersection of classical statistics, artificial intelligence (AI), and machine learning. The longest of these three lines is classical statistics. Without statistics, there would be no data mining, as statistics lays the foundation of most technologies on which data mining is built upon. Classical statistics embrace concepts such as regression analysis, standard distribution, standard deviation, standard variance, cluster analysis, and confidence intervals, all of which are used to study data and their relationships. [6] Those concepts are the very building blocks that helped bring about more advanced statistical analysis methods. Within the heart of today's data mining tools and techniques, classical statistical analysis plays a significant role.

Data mining's second longest family line is artificial intelligence. This discipline, which is built upon heuristics as opposed to statistics, attempts to apply human thought-like processing to statistical problems. Because this approach requires vast computer processing power, it was not practical until the early 1980s, when computers began to

offer useful power at reasonable prices. AI found a few applications at the very high end scientific/government markets, but the required supercomputers of the era priced AI out of the reach of virtually everyone else. [7] The notable exceptions were certain AI concepts which were adopted by some high-end commercial products, such as query optimization modules for Relational Database Management Systems. [6]

The third family line of data mining is machine learning, which is more accurately described as the union of statistics and AI. While AI was not a commercial success, its techniques were largely co-opted by machine learning. As Figure 2 shows below, the technological revolution increased computational ability while decreasing hardware and software costs. Machine learning was able to take advantage of the ever-improving price/performance ratios offered by computers of the 1980s and 1990s and it found more applications because the entry price was much lower than AI. Machine learning could be considered an evolution of AI, because it blends AI heuristics with advanced statistical analysis. Machine learning attempts to let computer programs learn about the data they study, allowing them to adapt their actions accordingly to previous results. Machine learning is used to analyze imprecise, incomplete, and complex information and deduct or find important relationships or patterns.

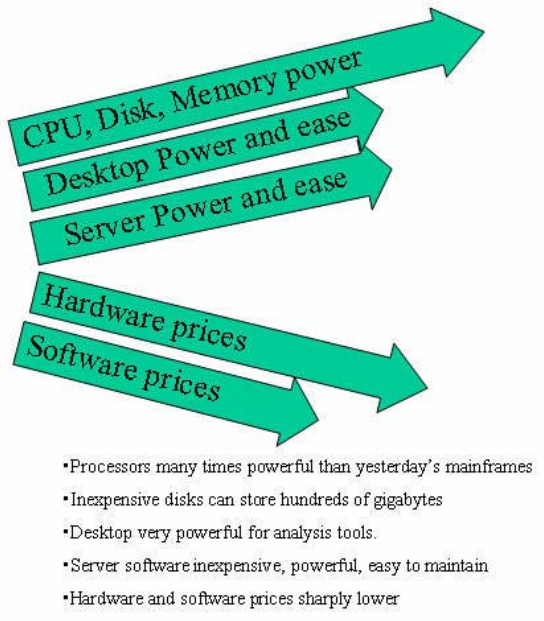


Figure 2. Impact of technological revolution. [From 8]

D. WHY DATA MINING EXISTS TODAY

As mentioned earlier, so much data is collected today. Many databases are now quantified in terabytes of data. These databases contain valuable information on customers, suppliers, employees, research publications, and general economic conditions. Substantial effort is required to turn the vast amount of information into something understandable. Finding some sort of meaning may be the difference between success and failure for many businesses. Additionally, the amount of complexity of data in corporate databases continues to grow at unprecedented rates and without data mining techniques and tools the ability of the analysts and managers to act and make strategic decisions is severely limited.

E. APPLICATIONS OF DATA MINING

Today, there are many applications of data mining. Figure 3 below shows the associated algorithms of each data mining application: information discovery, automated prediction of trends and behaviors and forensic analysis.

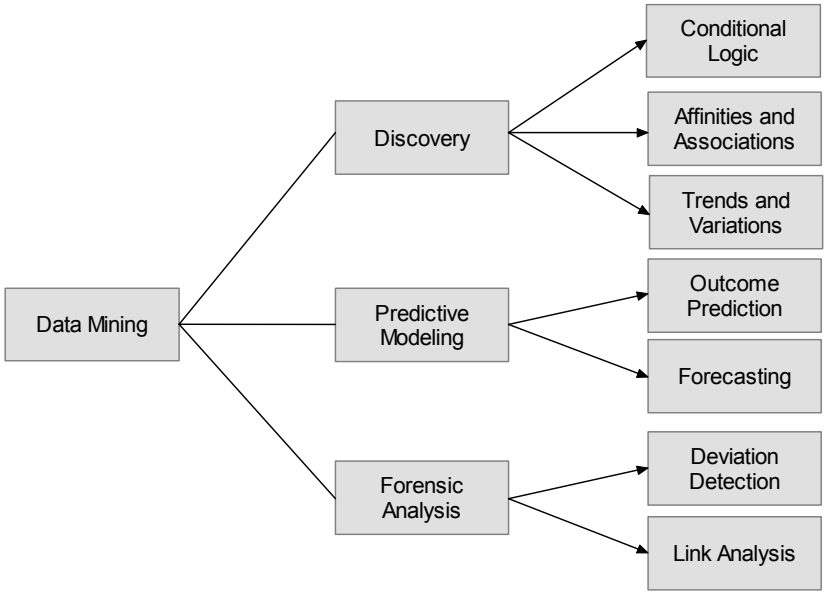


Figure 3. Data mining applications and associated algorithms. [From 9]

Data mining "discovery" is the process of identifying new patterns without specific direction as to the exact determining factors. This is a time consuming process requiring multiple iterations as the search for patterns is not focused and may uncover spurious relationships. However, it is a very powerful method to identify current relationships. This "discovery" mining is especially useful in decoding human gene sequences.

The second form is the automated prediction of trends and behaviors. In "predictive modeling," the data mining tool identifies patterns and uses that information to predict the future. This data mining application is particularly popular in customer retention programs, by segmenting the customer base, determining the most profitable category of customer, and predicting the ability to retain that customer. Using such data, a targeted program can be implemented to increase the probability of retaining the most profitable customer.

The third form of data mining, known as "forensic analysis," entails applying a pattern to the data (which may have been determined through a previous data mining exercise) in order to identify data anomalies. This application of data mining is prevalently used today by quality assurance and credit card fraud systems.

F. DATA MINING PROCESS

Every set of products must have a defined process behind it. To discover knowledge by finding trend and patterns, predicting behavior, or finding anomalies in a series of data, data mining follows a multi-step process. These steps are shown below as the Knowledge Discovery from Databases (KDD) process in Figure 4.

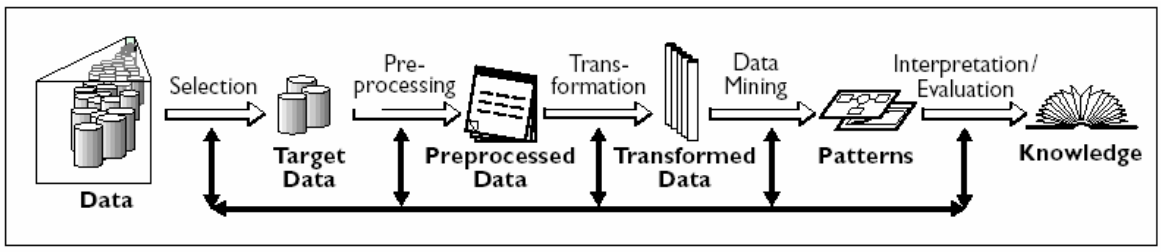


Figure 4. Overview of the steps constituting the KDD process. [From 10]

1. Obtain Data

The first step in the data mining process is the acquisition of data. The repositories of required data are usually disparate systems that include legacy databases, non-legacy databases, third party data sources (i.e. supplier databases), and even real-time transactions not currently stored. This data is then passed along to a central information repository called a data warehouse. At the data warehouse, the data is processed in the following ways:

1. The data is translated to a format that the data warehouse comprehends.
2. The data is cleansed to correct blank or invalid fields through a set of business rules which, for instance, may correct a missing zip code by marking the attribute as "blank".
3. The data is transformed to a standardized format, which for instance might translate gender data from one source of "M" to the standardized format of "male".

Once the data is processed, it is entered into the data warehouse and other respective data repositories

2. Focus Data

The second step involves selecting the target data through data reduction. This includes finding useful features to represent the data, depending on the goal of the task, and using dimensionality reduction or transformation methods to reduce the effective number of variables under consideration or to find invariant representations for the data. [10]

During this step, the data mining algorithm is determined (such as, summarization, classification, regression, and clustering), and the purpose of the model is derived.

3. Apply Data Mining Activities

With reducing the data and determining the algorithm to be used, the data mining tool will begin its work of clustering, classifying and scoring records. In clustering, the data mining tool identifies distinct clusters, or groupings, of similar data records. Using

these clusters, the data mining tool then classifies or categorizes each record into a cluster and assigns a score indicating the probability of the record fitting into and exhibiting the behavior of the identified cluster.

4. Learn from Results

The final step involves interpreting the results and using the knowledge acquired. The initiator of the process must translate the findings into meaningful action. In the case of a learning system, a hybrid system that combines artificial intelligence with data mining to "learn", the system incorporates the findings into its data warehouse to be used for future mining and prediction activities. In the case of an automated process, such as an e-commerce site, the findings may be used to present a specific product offering or discount to a current site visitor and potential customer. Similarly, if a degradation pattern is identified in an automated production quality monitoring and prediction system, the system might auto-alert the responsible production manager via e-mail or a page. If, however, the initiator is a business analyst, it is his/her responsibility to translate the discovered insight into meaningful action. The resulting patterns themselves add no value to the business or organization. Only changes enacted as a result of identifying and interpreting these data patterns and relationships add such value.

G. NOT A PANACEA

As hype regarding the promise of data mining grows in the business community, it is important to manage the expectations surrounding its immediate results. Companies must educate their consumers that data mining deals with human behavior and human interpretation of its results. [11] Other limitations include high costs to mine information, limited usability of expensive software tools, and the lack of efficient algorithms to search gigantic databases.

H. GATHERING & ANALYZING THE DATA

The raw datasets processed in this thesis were obtained through several online databases (e.g., IEEE's INSPEC, U.S. Department of Commerce's National Technical

Information Service). A sample of approximately 48,000 records was collected, each record represented abstracts centered on the following key words:

- Software Engineering
- Abstract Data Types
- Ada
- Java
- Rate Monotonic Analysis
- Software Cost Models
- Software Work Breakdown Structures
- Software Technology Transfer

The data in raw form, shown in Figure 5, was delimited text with field indicators. Appendix F contains a list and description of the data fields for the INSPEC database, which is where most of the data came from. All of the databases had similar formats for raw data making them compatible. The author analyzed the raw records and selected the fields necessary to carry out temperature-entropy calculations. The fields chosen were the following:

- Accession Number
- Affiliation
- Author
- Descriptors
- Identifier
- Language
- Number of References
- Source
- Publication Year
- Title

Record 5001 of 6023 in INSPEC 1985-1989

TI: On the fringe of Ada
AU: McMahon-PE
AA: CAE-Link Corp., Binghamton, NY, USA
SO: Proceedings of the IEEE 1989 National Aerospace and Electronics Conference NAECON 1989 (Cat. No.89CH2759-9). IEEE, New York, NY, USA; 1989; 4 vol. 2102 pp.
p.484-90 vol.2
PY: 1989
RT: Conference-Paper
CD: 22-26 May 1989; Dayton, OH, USA. Sponsored by: IEEE
CP: USA
LA: English
AB: The author compares features of Ada with that of Fortran in a real-time simulation environment, providing insight into potential problem areas on early Ada programs. Real-time software architectures in Ada and Fortran are compared, with a focus on potential impacts due to Ada's complexity...
RF: 2
DE: Ada-; digital-simulation; real-time-systems; software-engineering
ID: data-management; real-time-simulation-environment; Ada-; Fortran-; Ada-compilers; simulation-software; Ada-problem-areas; risk-reduction-strategies; target-computer-limitations
CC: C6110B (Software-engineering-techniques); C0310F (Software-development-management); C61; C03; C6; C0
TR: G (General-or-Review)
CL: CH2759-9/89/0000-0484$01.00
SK: 1000000000019890000000000000000000000484
AN: 3496502
UD: 8900

Figure 5. Raw record extracted from INSPEC database.

By following the data mining process described earlier, the next step after selecting the data fields was to represent them along with their data in a structured form. The implementation of this step is described within Iteration One in Chapter III.

THIS PAGE INTENTIONALLY LEFT BLANK

III. SOFTWARE & METHODOLOGY DEVELOPMENT

A. SOFTWARE DEVELOPMENT PROCESS

The data analysis process and software were created using a prototyping and iterative design process.

Prototyping and iterative design enable us to create vastly improved products in less time than with the waterfall methodology, provided that the role of prototyping is well understood, and that it is managed properly (Wilson). Additionally, in this approach, development is organized into a series of short, fixed-length mini-projects called iterations; the outcomes of each is a tested, integrated, and executable system. [12]

The advantages to this approach as described by Wilson are the following:

- Early visualization of the product
- Crisp definition of requirements
- Early user testing
- Enhanced communication within the development organization
- Enhanced feedback to users.

The stages of the iterative/prototyping process are shown in Figure 6 and described below. The four stages have activities that map directly to the set of activities performed by a team of systems engineers when they set out to create a computer system solution. These activities remain fairly consistent from one project to another. [13] The activities are also discussed below under each iteration stage.

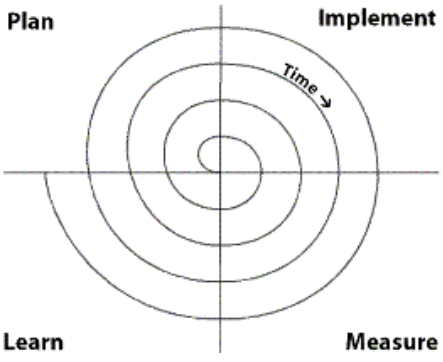


Figure 6. Iterative Development Cycle. [From 15]

1. Plan

The purpose of the planning stage is to determine the user's needs and figure out how those needs will be addressed. Planning encompasses the activities of requirements gathering, analysis and design.

a. Requirements Gathering

Requirements gathering is the task of documenting the functions that the application should perform for the users in the users' language and from the users' perspective.

Users view computer systems as black boxes meaning users care very little about the inner-workings of the computer system. Users want to be able to provide input to the system and obtain the output they expect. With this in mind requirements documentation that puts everything in the context of "going in" and "coming out" has more relevance to the users who study it. [13]

b. Analysis

Analysis encompasses the building of a logical solution that satisfies the requirements but does not necessarily take the physical constraints into account.

c. Design

Design takes the logical solution from the analysis and changes it to work effectively with the physical constraints (storage, database performance, caching, execution time, and so forth). The final output of design is a set of specifications that can direct the construction effort.

2. Implement

During implementation, a prototype is built to test the solutions developed during the planning stage. The activity, construction, is seen here.

a. Construction

Construction uses the design to produce working code, which involves making the lowest-level design decisions, writing code, compiling, debugging, and testing by increment. [13]

3. Measure

Wilson points out that measuring involves gauging user interaction through a series of questions. Such questions are the following:

- How long does it take users to understand the solution?
- How long does it take them to do their work?
- What problems do they encounter?

These measurements must be both subjective and objective; time on task is important, but if a user takes more time but produces substantially higher quality work, then the weight of the improved quality needs to be considered.

a. Testing

Testing is the activity of using the constructed application to produce a complete working system by system testing, detecting and recording issues, fixing problems, and getting user acceptance of the result.

4. Learn

During the learning phase it is decided which parts of the prototype are working and what parts are not.

There are many reasons a prototype does not achieve its goals, the most significant reason being an incorrect understanding of what the users want to accomplish (requirements).

Iterative development proves to be the best software development process for this type of project due to the fact that all of the user's needs were not known before the project started. Iterative development is based on an attitude of embracing change and adaptation as unavoidable and indeed essential drivers. [12]

Wilson explains that as development cycles through the phases, the developers are gaining a fuller understanding of the users and their needs, and coming closer to a finished product. The development cycles through these four phases until there is an agreed upon satisfaction between the developers and the users obtained through the measure and learning phases.

The data analysis tool presented in this thesis went through four iterations before it could be publicly released. Each iteration is discussed in detail within the following sections.

The requirements for each iteration were gathered from a series of interviews between the developer (this thesis's author) and the primary researcher (Dr. Michael Saboe). These interviews are presented in Appendix A.

It is also important to note that for the first three iterations the primary user of the developed applications was the developer himself. It is not until iteration four that DataThermometer is born and the user base switches.

B. ITERATION ONE: ESTABLISHING THE PROJECT

Iteration one marked the beginning of the software development portion of the project. The first iteration was where the first set of requirements was gathered and where the first prototype was developed. The requirements for this iteration came from the first interview found in Appendix A.

1. Plan

From the requirements interview, the goals of this iteration were to fulfill the requirements listed below:

- The system shall parse user supplied data from INSPEC databases.
- The system shall remove duplicate records from the data.
- The system shall standardize the data fields.
- The system shall compute a time-based entropy calculation based on the descriptors field and the publication year of the data.

The goals of this iteration are also to satisfy the first two steps of the data mining process discussed in Chapter II (Obtain and Focus data).

The use case diagram, Figure 7 below, shows the functionality of this iteration that must be available to the data analyst.

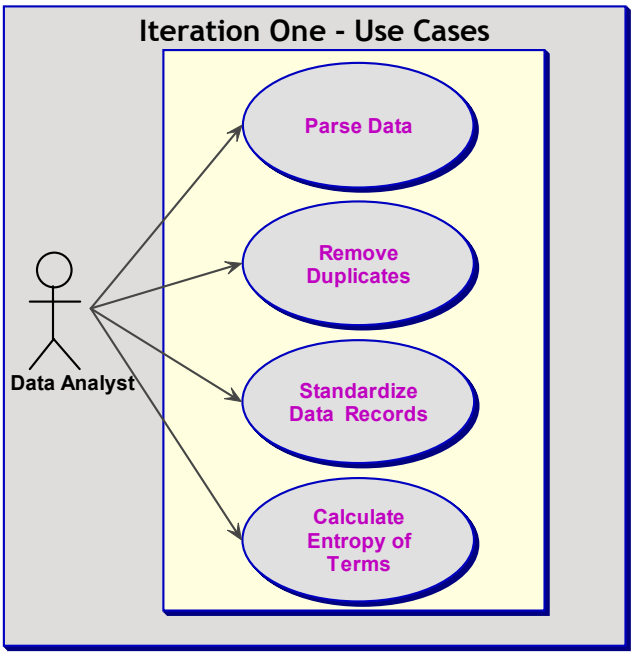


Figure 7. Iteration one use cases.

Two commercial off the shelf (COTS) applications facilitated in meeting the above requirements: The Technology Opportunities Analysis of Scientific Information System (Tech OASIS) and Microsoft Excel (MS Excel).

Tech OASIS is a government-tailored software program that is commercially available under the trade name VantagePoint. One of the many uses of Tech OASIS is ability to process raw bibliographic data through the use of import filters. Once the data is imported, Tech OASIS can pre-process the data to remove duplicates and standardize it. Additionally, Tech OASIS can be extended by using the built in script editor for custom analysis.

Figure 8, below shows how a user would use Tech OASIS to process data obtained from bibliographic databases.

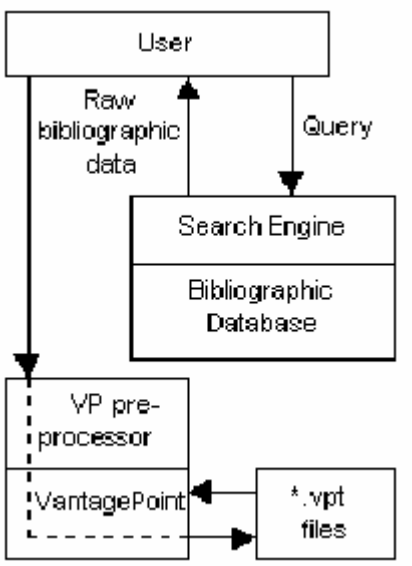


Figure 8. Tech OASIS data file creation.

Microsoft Excel was used to generate the entropy charts. Tech OASIS interfaced with MS Excel by using the visual basic scripting feature. A script in Tech OASIS sends data to MS Excel and applies MS Excel functions on that data in the form of mathematic formulas. Once results are obtained the script runs MS Excel functions relating to chart generation.

Below, Figure 9, is a class diagram based on the analysis. The class diagram shows Tech OASIS has import filters, export scripts, and data processing methods. The export script is used to bridge the gap between MS Excel and Tech OASIS.

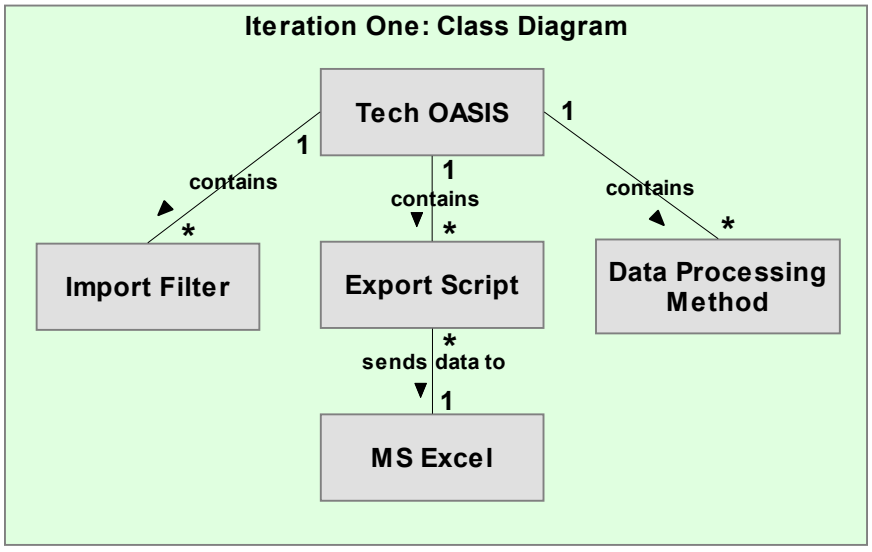


Figure 9. Iteration one class diagram.

The sequence diagram below (Figure 10) explains how a use would interface with this setup.

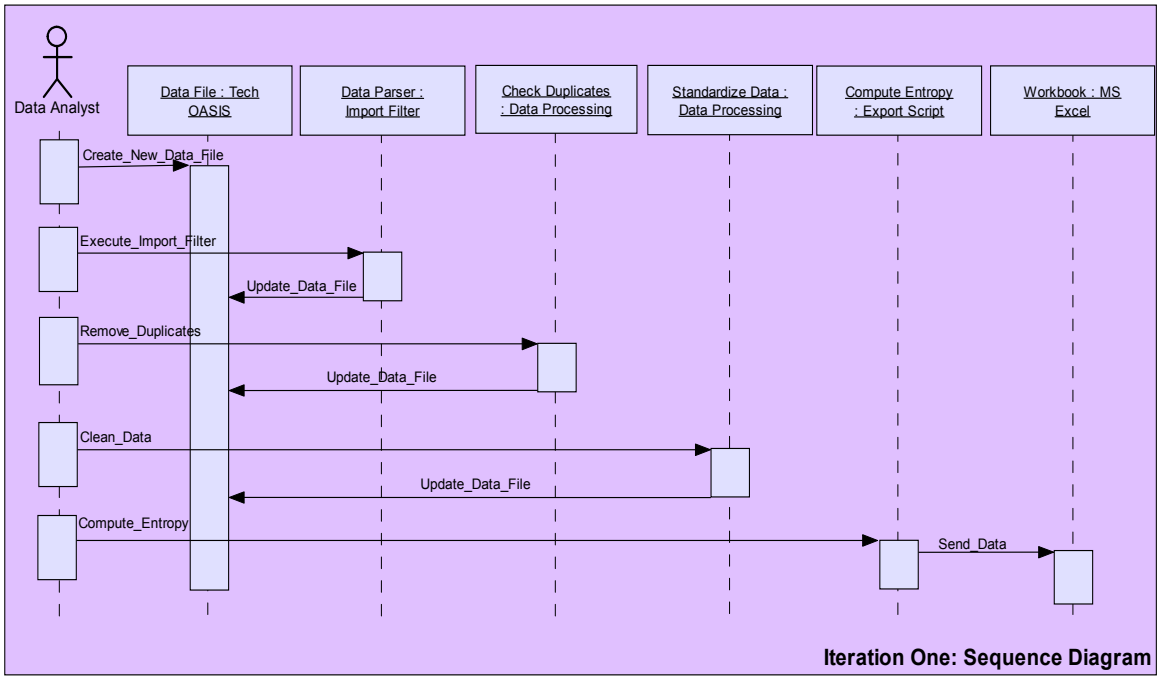


Figure 10. Iteration one sequence diagram.

The following steps can be seen in the sequence diagram shown in Figure 10:

1. The data analyst creates a new data file.
2. The data analyst runs an import filter to parse the data.
3. The data parser object updates the data file.
4. The data analyst runs the remove duplicates command.
5. The data file is updated.
6. The data analyst runs the clean data command.
7. The data file is updated.
8. The data analyst executes the compute entropy script.
9. The export script sends data to MS Excel.

2. Implement

The implementation consisted of creating an import filter to extract the data from its raw form and place it in a Tech OASIS data file.

a. Importing the Data

Importing the data required the use of an import filter to decipher the raw data and place it into a structured form. The author created a new database configuration file within Tech OASIS where the filter syntax was placed. Figure 11 shows the *Database Configuration Editor* with the finished import filter. After completing the filter, the data was ran through and organized into a structured representation in Tech OASIS. Several activities were then performed on the data to prepare it for the entropy analysis.

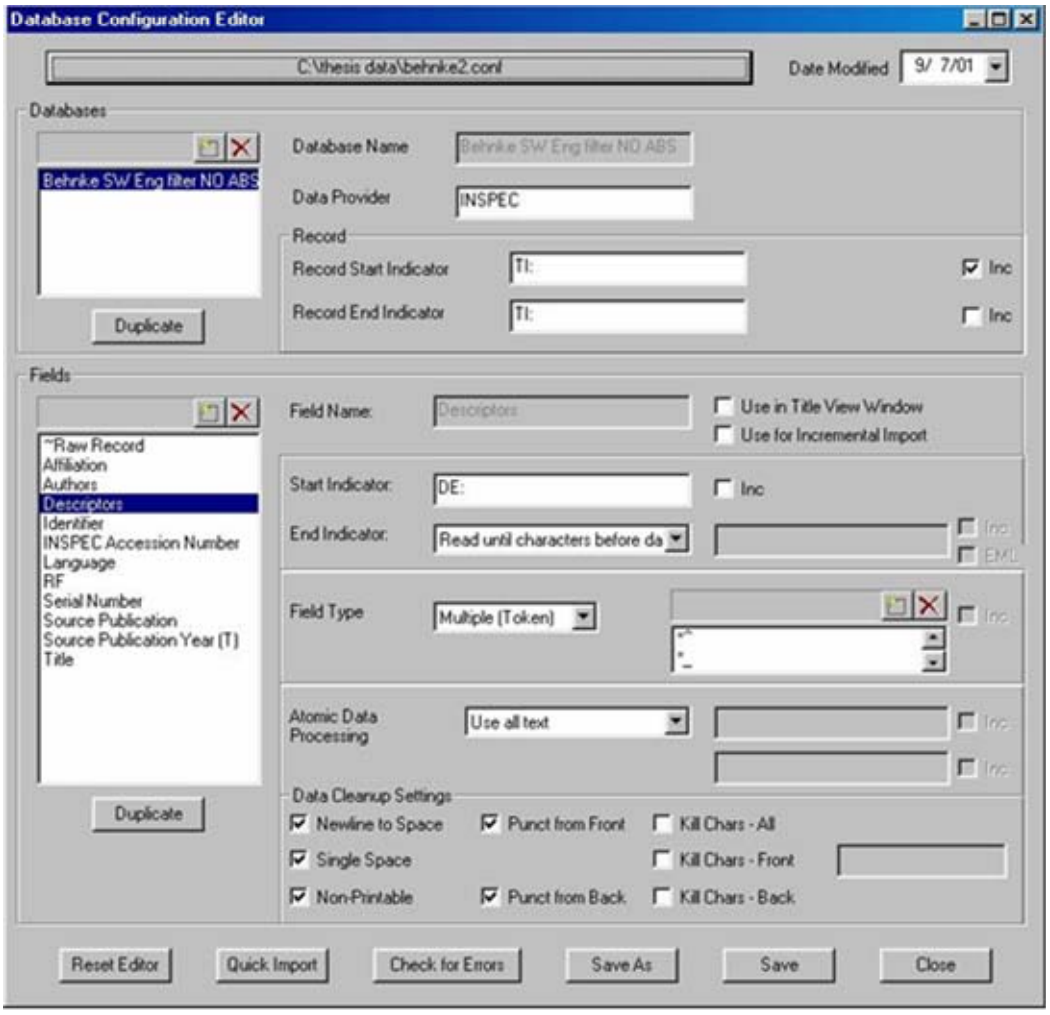


Figure 11. Database Configuration Editor showing the import filter.

b. Preparing the Data

Finishing touches were then applied to the data to prepare it for export. In a discussion in August with Robert Watts, he recommended that the dataset be “cleaned” before performing calculations on it. Tech OASIS cleans a list by attempting to identify list items that may be equivalent. For example, the terms "human-computer interaction" and "human computer interaction" will appear as separate items on a list (because of the hyphen between "human" and "computer" in the first instance). The list cleanup algorithms in Tech OASIS will catch this as well as plurals and simple misspellings. In addition, Tech OASIS can identify equivalents such as J. Smith, James Smith, and Smith, J. Figure 12 shows the fields that were cleaned. Note the different number of instances

between the cleaned and non-cleaned fields. In addition to cleaning the fields, duplicate accession numbers were removed. An accession number is a unique number assigned to each record.

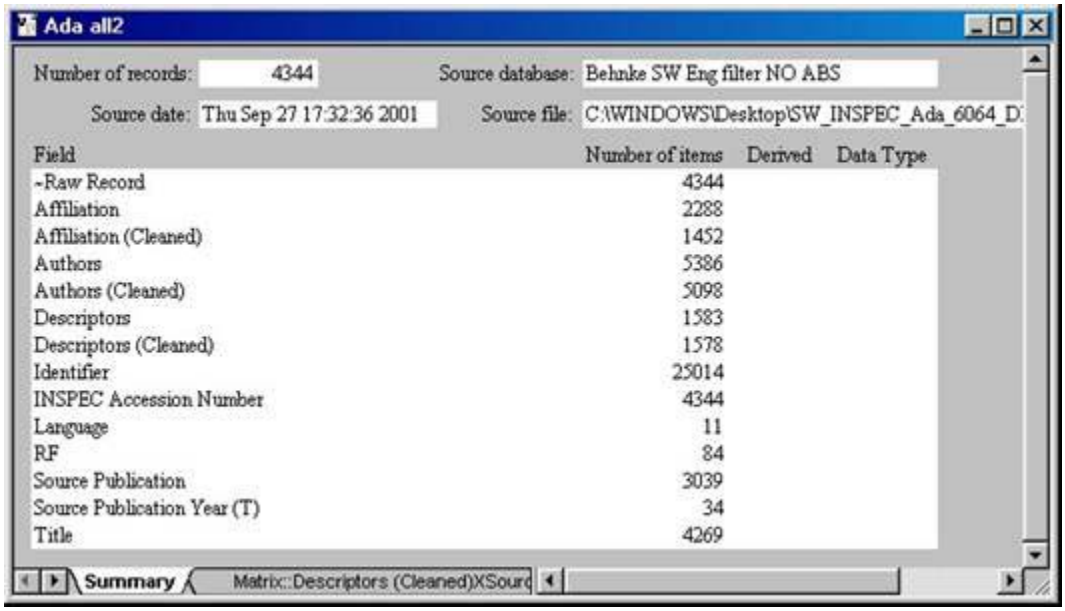


Figure 12. Ada dataset summary that shows cleaned fields.

Before duplicates were removed from the datasets there were ~48,000 records, afterwards there were 22,219. The number of records for each dataset is shown in Table 2.

Dataset	**Records**	**Descriptors**	**Publication Years**
Software Engineering	10,763	2425	1978-2000
Abstract Data Types	567	364	1974-2000
Ada	4195	1460	1979-2000
Java	5906	2421	1995-2000
Rate Monotonic Analysis	223	342	1986-2000
Software Cost Models	273	394	1972-2000
Software Work Breakdown Structures	36	63	1978-2000
Software Technology Transfer	256	222	1982-2000
Totals	*22,219*	*7691*	

Table 2. Number of records for each dataset.

c. Export Script and Entropy Calculation

After importing and preparing the data, an export script had to be constructed to calculate the entropy of the descriptors. The export script is where the user interacts with Tech OASIS by selecting the data and time fields used to calculate entropy. Figure 13 shows the script prompting the user to select the data and time fields, Figure 14 displays a list of fields that the user can select from, and Figure 15 presents what happens when the user selects a time field that doesn't contain groups (time intervals).

Figure 13. Tech OASIS script prompts user to make a selection.

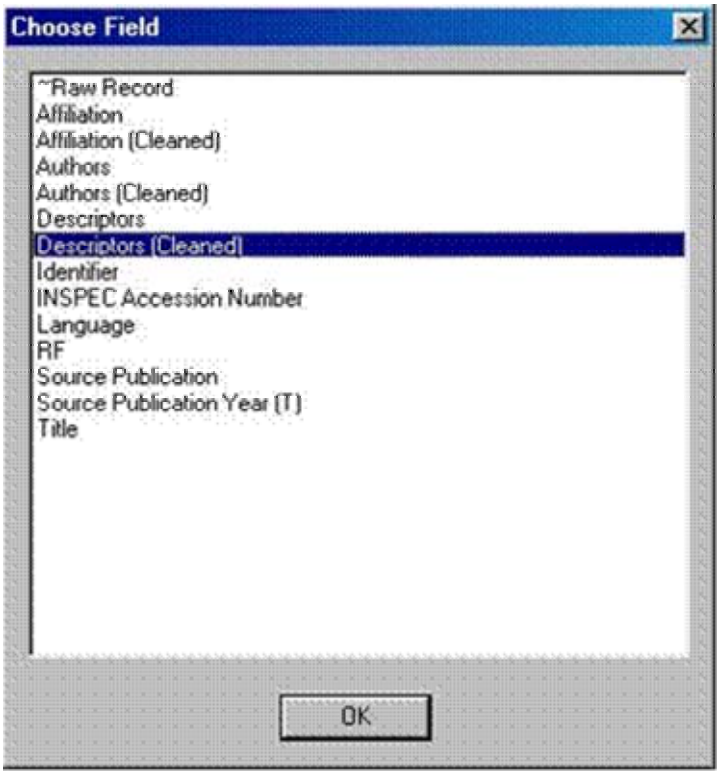


Figure 14. User selection screen in Tech OASIS.

Figure 15. Error message when time field has no groups.

The script creates a co-occurrence matrix of the data field (rows) and the time field (columns) and exports it into Microsoft Excel. Then the export script finished the entropy calculations and generated the charts. Figure 16 shows a portion of the co-occurrence matrix created in Tech OASIS. Figure 17 presents the exported data in Microsoft Excel.

		Descriptors (Cleaned)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
		# Records	0	108	142	146	238	178	268	288	49	290	156	137	309	258	273	257	211	281	188	188	169
Source Publication Year (1)	# Records	1	1979	1980	1981	1982	1983	1984	1985	1986	1987	1988	1989	1990	1991	1992	1993	1994	1995	1996	1997	1998	1999
1	2697	Ada		48	86	89	126	79	149	170	23	198	93	107	217	176	196	163	141	192	129	113	120
2	548	software-engineering			14	20	56	33	33	55	13	73	33	21	45	25	27	26	20	20	8	7	3
3	542	real-time-systems					10	2	9	17	3	28	20	24	59	54	49	49	42	62	41	26	31
4	496	object-oriented-programming						1			2	11	18	30	39	33	37	44	34	67	38	52	65
5	395	software-tools							19	33	6	42	18	24	42	30	25	34	26	39	24	13	5
6	389	program-compilers		10	10	14	31	18	25	28	6	29	18	11	26	21	26	28	15	19	13	11	19
7	280	object-oriented-languages													1	10	19	17	31	43	31	43	59
8	276	programming-environments							48	32	11	28	19	11	26	12	13	14	9	25	11	5	9
9	274	software-reusability									2	9	9	23	37	23	33	22	25	27	30	18	7
10	255	military-computing		1	4	1	13	2	13	22	8	19	9	16	34	20	30	16	9	16	9	4	5
11	236	programming		4	4	6	24	16	13	28	5	25	16	1	18	18	12	11	15	9	4	4	1
12	233	parallel-programming							3	6	5	14	14	13	25	21	11	16	15	35	18	21	6
13	228	high-level-languages		7	27	19	38	30	23	24	2	12	8		7	8	2	6	4	4	3	1	2
14	219	aerospace-computing		1	2		1		4	7		20	4	19	26	21	16	27	10	21	15	7	10
15	196	formal-specification									1	4	8	9	19	27	18	21	18	22	15	13	12
16	186	computer-science-education				2	6	6	6	9	2	29	8	3	9	11	11	10	7	15	10	27	7
17	182	distributed-processing		1	1	5	8	3	17	13	1	20	12	7	14	15	11	6	12	6	15	7	5
18	177	data-structures		1	6	6	4	4	18	19	6	28	12	7	11	14	12	1	5	5	4	10	2
19	164	program-testing		3	1	1	3	1	3	7	1	6	9	7	14	13	11	13	13	23	9	11	8
20	147	digital-simulation			4	5	5	33	5	10	1	8	3	7	6	6	11	8	8	11	7	5	3
21	140	software-portability		1	6	2	5	4	1	6	1	10	1	4	10	10	12	12	13	11	8	11	9
22	128	program-verification							5	6	1	7	4	8	11	6	12	10	7	19	9	5	8
23	118	software-reliability							1	7	1	8	5	8	11	13	16	12	2	12	5	4	4
24	109	operating-systems-computers		3	5	6	7	5	8	15	1	6	7	4	6	6	11	10	1	2	3	2	

Figure 16. Co-occurrence matrix created in Tech OASIS.

	Time step	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	# Records	8	108	142	146	248	170	268	288	49	290	156	137	309	258	273
# Records	Descriptors (Cleaned)	1979	1980	1981	1982	1983	1984	1985	1986	1987	1988	1989	1990	1991	1992	1993
2697	Ada		48	86	89	128	79	149	170	23	198	93	107	217	176	196
548	software-engineering			14	20	56	33	33	55	13	73	33	21	45	25	27
542	real-time-systems					10	2	9	17	3	28	20	24	59	54	49
496	object-oriented-programming						1			2	11	18	30	39	33	37
395	software-tools							19	33	6	42	18	24	42	30	25
389	program-compilers		10	10	14	31	18	25	28	6	29	18	11	26	21	26
280	object-oriented-languages													1	10	19
276	programming-environments							48	32	11	28	19	11	26	12	13
274	software-reusability									2	9	9	23	37	23	33
255	military-computing		1	4	1	13	2	13	22	8	19	9	16	34	20	30
236	programming		4	4	6	24	16	13	28	5	25	16	1	18	18	12
233	parallel-programming							3	8	5	14	14	13	25	21	11
228	high-level-languages		7	27	19	38	30	23	24	2	12	8		7	8	2
219	aerospace-computing		1	2		1		4	7		20	4	19	26	21	16
196	formal-specification									1	4	8	9	19	27	18
186	computer-science-education				2	6	6	6	9	2	29	8	3	9	11	11
182	distributed-processing		1	1	5	8	3	17	13	1	20	12	7	14	15	11
177	data-structures		1	6	6	4	4	18	19	6	28	12	7	11	14	12
164	program-testing		3	1	1	3	1	3	7	1	6	9	7	14	13	11
147	digital-simulation			4	5	5	33	5	10	1	8	3	7	6	6	11
140	software-portability		1	6	2	5	4	1	6	1	10	1	4	10	10	12
128	program-verification							5	6	1	7	4	8	11	6	12
118	software-reliability							1	7	1	8	5	8	11	13	16
109	operating-systems-computers		3	5	6	7	5	8	15	1	6	7	4	6	6	11

Figure 17. Exported data in Microsoft Excel.

The first iteration of the entropy calculation method was very simple in comparison to the methods of later iterations. The basic operations included calculating the cumulative entropy of the user-selected data field and creating a graph based on the results of the calculation. The cumulative entropy of the data field was calculated by applying the formula, given by Dr. Michael Saboe, to the exported co-occurrence matrix. The cumulative entropy formula and example are shown in Appendix B. Figure 18 shows a sample of the data after the cumulative entropy formula has been applied. The resultant chart based on the cumulative entropy calculated is presented as Figure 19.

	Time step	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	# Records	8	108	142	146	248	170	268	288	49	290	156	137	309	258	273
# Records	Descriptors (Cleaned)	1979	1980	1981	1982	1983	1984	1985	1986	1987	1988	1989	1990	1991	1992	1993
2697	Ada	0	0.46732	0.4905	0.49352	0.4886	0.48321	0.47697	0.46707	0.46352	0.45934	0.45541	0.45203	0.44742	0.44172	0.43957
548	software-engineering	0	0	0.13101	0.17413	0.24177	0.25372	0.23823	0.23615	0.23758	0.24227	0.24102	0.23289	0.22175	0.20898	0.19968
542	real-time-systems	0	0	0	0	0.04776	0.0457	0.05466	0.06801	0.06938	0.0854	0.09511	0.10446	0.12348	0.1351	0.14178
496	object-oriented-programming	0	0	0	0	0	0.00567	0.00424	0.00321	0.00799	0.02378	0.04265	0.06588	0.08269	0.09152	0.0998
395	software-tools	0	0	0	0	0	0	0.05047	0.08669	0.09066	0.11305	0.11751	0.1238	0.12974	0.12983	0.12802
389	program-compilers	0	0.19325	0.16916	0.17413	0.19473	0.1957	0.18623	0.17557	0.17468	0.16678	0.16523	0.1586	0.15056	0.14408	0.14104
280	object-oriented-languages	0	0	0	0	0	0	0	0	0	0	0	0	0.00169	0.01199	0.02498
276	programming-environments	0	0	0	0	0	0	0.10364	0.11988	0.127	0.12854	0.13185	0.12825	0.12506	0.11756	0.11223
274	software-reusability	0	0	0	0	0	0	0	0	0.00563	0.01945	0.02908	0.04931	0.06903	0.07476	0.08383
255	military-computing	0	0.0334	0.05977	0.04682	0.07914	0.07117	0.0797	0.09173	0.0977	0.0983	0.09753	0.10096	0.1071	0.10563	0.10887
236	programming	0	0.09971	0.08615	0.09091	0.13324	0.14474	0.13269	0.13598	0.13606	0.13324	0.13393	0.1238	0.11705	0.11333	0.10802
233	parallel-programming	0	0	0	0	0	0	0.01094	0.02502	0.03267	0.04436	0.05502	0.06167	0.07037	0.07476	0.07315
228	high-level-languages	0	0.15054	0.24207	0.23505	0.2435	0.25108	0.22609	0.203	0.19775	0.17423	0.16585	0.15291	0.13636	0.1258	0.11514
219	aerospace-computing	0	0.0334	0.03973	0.02663	0.02259	0.01853	0.02496	0.0323	0.03099	0.05019	0.04998	0.06252	0.0717	0.07589	0.07661
196	formal-specification	0	0	0	0	0	0	0	0	0.00307	0.00998	0.02035	0.02862	0.04057	0.05396	0.05865
186	computer-science-education	0	0	0	0.019	0.03991	0.0517	0.05258	0.05497	0.05568	0.07669	0.07739	0.07323	0.06903	0.06789	0.06707
182	distributed-processing	0	0.0334	0.02853	0.05296	0.06585	0.06308	0.0815	0.08152	0.07967	0.0854	0.08853	0.08642	0.08332	0.0825	0.08001
177	data-structures	0	0.0334	0.07774	0.0859	0.07261	0.07117	0.08855	0.09421	0.0977	0.10622	0.10696	0.10307	0.09552	0.09256	0.08943
164	program-testing	0	0.08006	0.05007	0.04043	0.03991	0.03622	0.03483	0.03916	0.03918	0.03952	0.04583	0.0484	0.0521	0.05459	0.05539
147	digital-simulation	0	0	0.05007	0.06459	0.06238	0.13087	0.11005	0.09909	0.09654	0.08729	0.08261	0.08109	0.07369	0.06905	0.0681
140	software-portability	0	0.0334	0.07774	0.06459	0.06238	0.06308	0.05047	0.04884	0.0484	0.05133	0.04792	0.04747	0.04835	0.04949	0.0515
128	program-verification	0	0	0	0	0	0	0.01686	0.02502	0.02579	0.0306	0.03261	0.03789	0.04137	0.04081	0.04401
118	software-reliability	0	0	0	0	0	0	0.00424	0.01919	0.02031	0.02793	0.03145	0.03689	0.04057	0.04488	0.0498
109	operating-systems-computers	0	0.08006	0.08615	0.09091	0.08548	0.08395	0.0797	0.08283	0.08092	0.07269	0.07294	0.07	0.06425	0.06076	0.06079

Figure 18. Data in Excel after applying entropy formula.

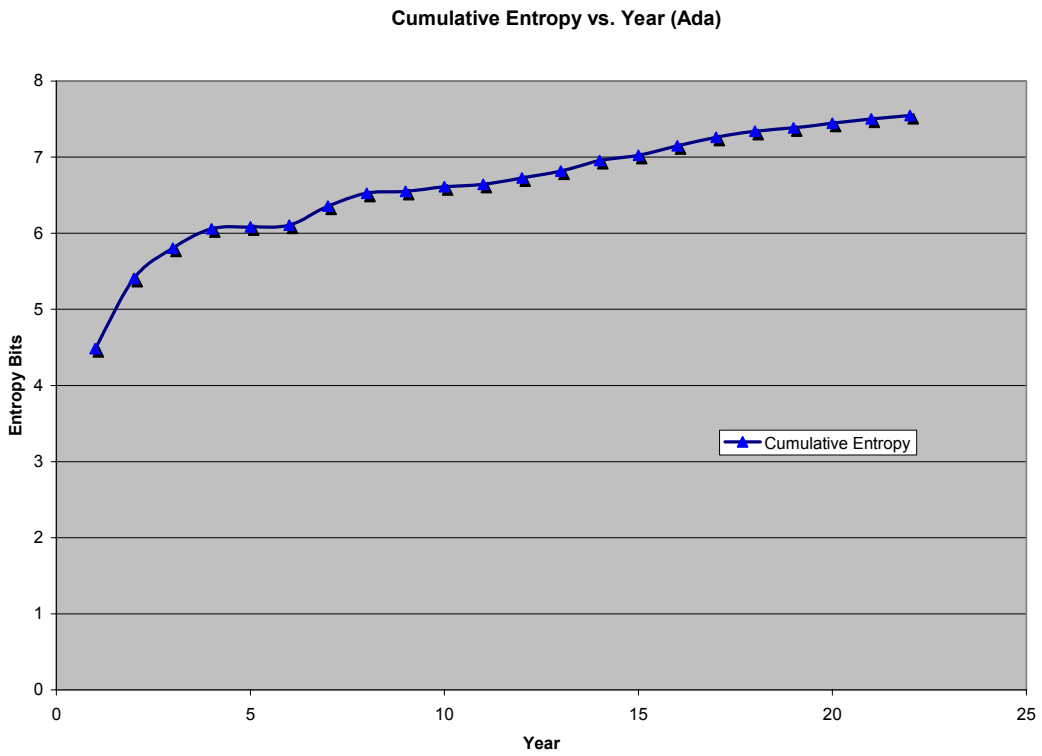


Figure 19. Cumulative entropy chart from iteration one.

3. Measure

Measuring the user's interactions was not important during this iteration because the user and the developer were the same person. The ability to perform this type of analysis in such a small time period provided the greatest benefit.

4. Learn

The initial set of requirements was fulfilled, but improvements and additional functionality were needed in the entropy analysis area.

C. ITERATION TWO: EXPANDED ENTROPY ANALYSIS

After iteration one, several requests for more functionality were received from Michael Saboe. As shown in the interviews from Appendix A, the first request asked for the addition of a power trend-line to the cumulative entropy chart. Second, a summary

sheet was requested. The desired features of the summary sheet were to display the following:

- Time Intervals
- Cumulative entropy in each time interval (slice)
- Percent difference of the cumulative entropy and the predicted value in a time interval. Predicted value comes from using the power trend-line's equation (see Appendix B under "Predicted Entropy Calculation").
- Predicted entropy values based on 5 years and 10 years of data

The third request involved implementing new formulas supplied by Dr. Saboe on a new worksheet titled "Entropy lambda" to calculate the derivative of the cumulative entropy's trend-line equation and lambda from using that calculated derivative. "Time Interval Derivative Calculation" and "Lambda Calculation" in Appendix B explain the formulas behind the new calculations. Along with the new calculations a graph was added to display cumulative entropy with lambda for each time interval and the difference of lambda subtracted from cumulative entropy.

A fourth request was to calculate the Lyapunov exponent. The Lyapunov exponent used with lambda gives another method of computing entropy. This request added more to the "entropy lambda" sheet and a new chart. The calculation of the Lyapunov exponent came from another fomula supplied by Dr. Saboe. "Lyapunov Exponent Calculation" in Appendix B shows the formula and an example of its use.

1. Plan

The requirements that were added during this iteration were the following:

- The system shall create an entropy summary sheet with the information specified above.
- The system shall add a power trendline to the entropy chart.
- The system shall format the entropy chart.
- The system shall create and populate a worksheet titled "Entropy Lambda".
- The system shall calculate the Lyapunov exponent.

The use cases in this iteration remain the same from iteration one. Figure 20 shows the revised class diagram. The class diagram reflects the addition of MS Excel macros written in Visual Basic for Applications (VBA).

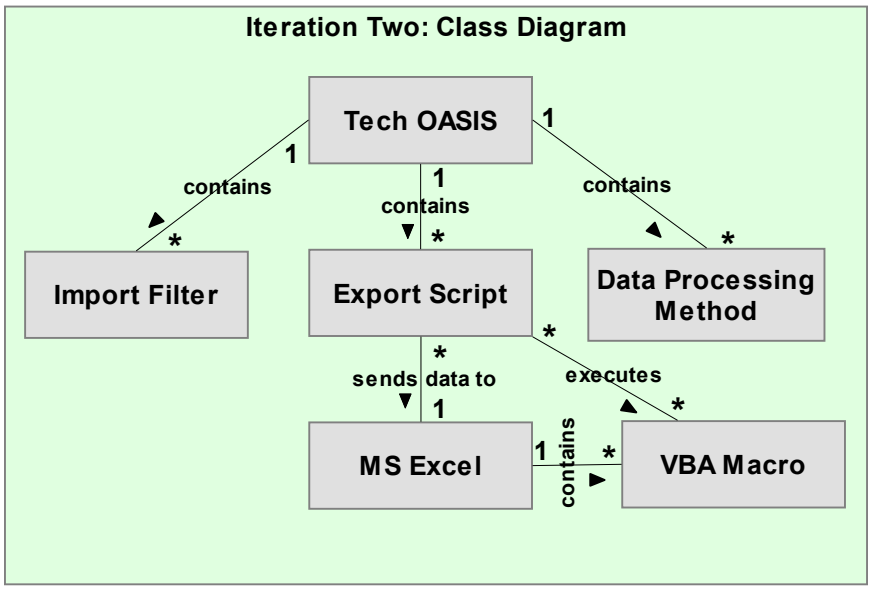


Figure 20. Iteration two class diagram

The sequence diagram for iteration two is shown in Figure 21. The sequence diagram also reflects the addition of the Excel macros.

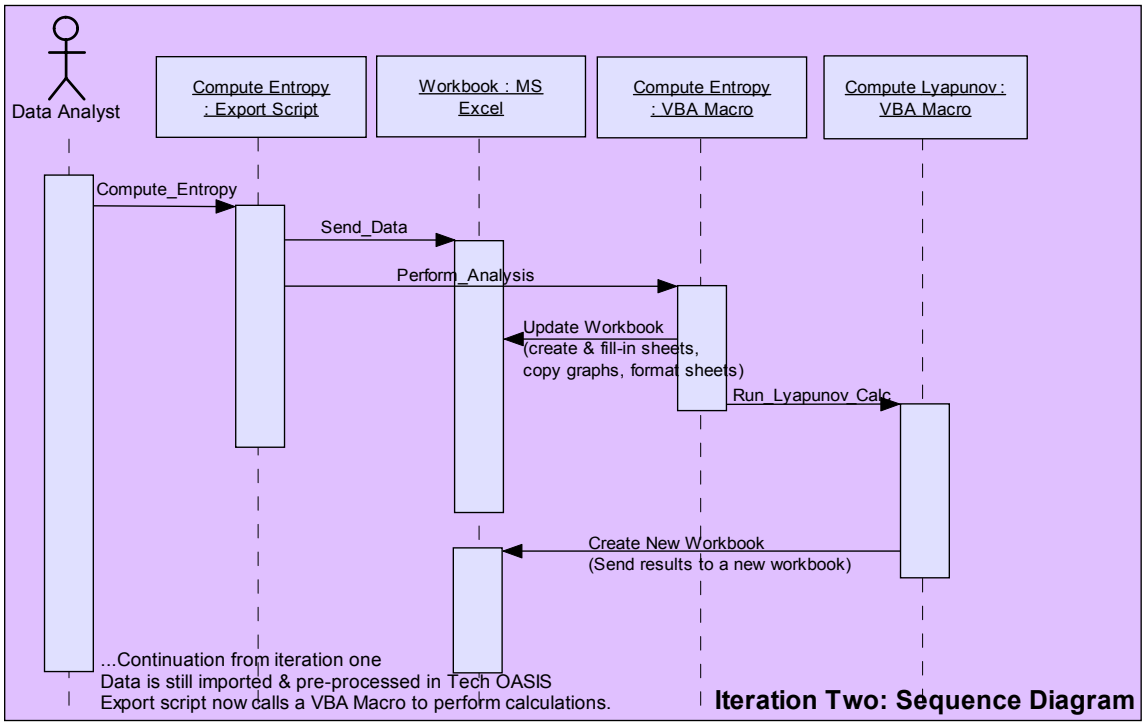


Figure 21. Iteration two sequence Diagram

The sequence diagram shows the following steps:

1. The Tech OASIS export script sends the data to MS Excel and then calls the Compute Entropy Excel macro.
2. The Compute Entropy macro performs the entropy computations, generates the cumulative entropy summary sheet and graph.
3. The Compute Entropy macro calls the Compute Lyapunov macro.
4. The Compute Lyapunov macro creates a new workbook to calculate and store the entropy lambda sheets and graphs as well as the Lyapunov exponent calculation results.

2. Implement

During this iteration, the entropy calculation activity was moved to a MS Excel macro to increase system modularity and improve scalability. Another macro was developed to create the entropy lambda sheet, compute the Lyapunov exponent, and display the results.

Figure 22 shows the revised cumulative entropy chart with the power trend-line and formatting. Figure 23 shows the generated summary sheet.

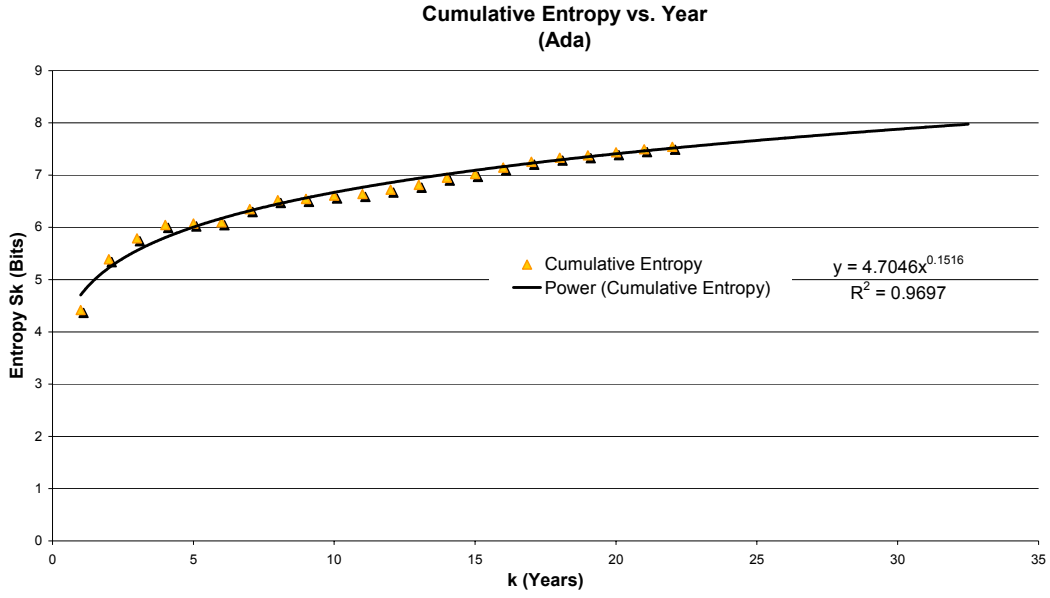


Figure 22. Cumulative entropy chart final revision with trend-line.

Time T	Slice	Cum Entropy	Error (Act vs. Pred) y = 4.7046x0.1516	Predicted 5 years of data	Predicted 10 years of data
1	1979	4.418295834	6.47%	4.418295834	4.418295834
2	1980	5.390283683	-3.10%	5.390283683	5.390283683
3	1981	5.79172667	-4.13%	5.79172667	5.79172667
4	1982	6.046115945	-4.08%	6.046115945	6.046115945
5	1983	6.073590809	-1.24%	6.073590809	6.073590809
6	1984	6.099329465	1.08%	6.165497867	6.099329465
7	1985	6.350240422	-0.82%	6.310693927	6.350240422
8	1986	6.522610058	-1.26%	6.439229457	6.522610058
9	1987	6.54504976	0.15%	6.554777164	6.54504976
10	1988	6.608194265	0.78%	6.659893941	6.608194265
11	1989	6.639861872	1.76%	6.756435086	6.756435086
12	1990	6.723235053	1.82%	6.845791711	6.845791711
13	1991	6.815179001	1.67%	6.929035067	6.929035067
14	1992	6.951253065	0.80%	7.007008534	7.007008534
15	1993	7.022179419	0.83%	7.080388584	7.080388584
16	1994	7.146156825	0.05%	7.149726525	7.149726525
17	1995	7.256909346	-0.57%	7.215477689	7.215477689
18	1996	7.336841465	-0.80%	7.278023631	7.278023631
19	1997	7.381119588	-0.59%	7.337685724	7.337685724
20	1998	7.440142251	-0.61%	7.39473887	7.39473887
21	1999	7.49800686	-0.62%	7.449419417	7.449419417
22	2000	7.541403763	-0.52%	7.501932253	7.501932253

Figure 23. Summary sheet generated by the macro.

The additional requests added more source code to the macro. The result of the additions of the formulas to calculate lambda and displaying the results is shown below. The "Entropy lambda" sheet is displayed in Figure 25. The graph of the results is presented in Figure 24.

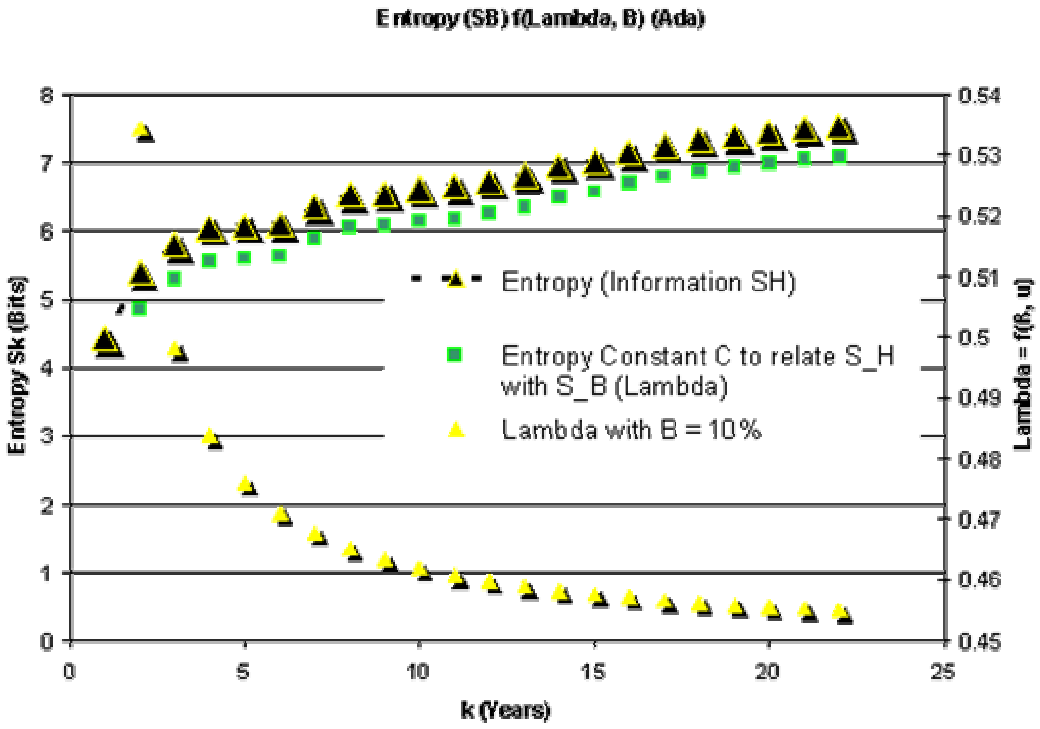


Figure 24. Cumulative entropy with lambda & difference.

The calculation of the Lyapunov exponent involved using the equation of a power trend-line from a new entropy chart. The new chart needed to show a map of $S_{H\,K}$, $S_{H\,K+1}$ where $S_{H\,K}$ is the cumulative entropy of time step k. The resulting additions to the entropy lambda sheet are shown below in Figure 25. Figure 26 shows the chart, the map of $S_{H\,K}$, $S_{H\,K+1}$, produced in this iteration.

Time T	Slice	Cum Entr	Error (Act vs. Pred) y = 4.7046x 0.1516	Cum_K+1	Exp J'(k,k+1) = .7122*1.7 6 k^(.7122-	du_(t-c)	du_(t)	du_(t-2)	du_(t-5)	du_(t-15)	du_(t-20)	C_y_10%	Lambda_E	B10%_y
1	1979	4.4183	6.47%	5.390284	0.817354									
2	1980	5.39028	-3.10%	5.791727	0.771892	0.710304	0.3943					4.855864	0.534419	0.1
3	1981	5.79173	-4.13%	6.046116	0.756099	0.394339	0.2795	0.7103				5.293538	0.498188	0.1
4	1982	6.04612	-4.08%	6.073591	0.746802	0.279491	0.2189	0.3943				5.562355	0.483761	0.1
5	1983	6.07359	-1.24%	6.099329	0.745829	0.218925	0.1811	0.2795				5.597626	0.475965	0.1
6	1984	6.09933	1.08%	6.35024	0.744921	0.181142	0.1552	0.2189	0.7103			5.628254	0.471076	0.1
7	1985	6.35024	-0.62%	6.52261	0.736329	0.155165	0.1361	0.1811	0.39434			5.88252	0.467721	0.1
8	1986	6.52261	-1.28%	6.54505	0.730675	0.136131	0.1215	0.1552	0.27949			6.057335	0.465275	0.1
9	1987	6.54505	0.15%	6.608194	0.729953	0.12154	0.11	0.1361	0.21892			6.081637	0.463413	0.1
10	1988	6.60819	0.78%	6.639862	0.727939	0.109975	0.1006	0.1215	0.18114			6.146247	0.461947	0.1
11	1989	6.63986	1.76%	6.723235	0.726938	0.100564	0.0927	0.11	0.15517			6.179098	0.460764	0.1
12	1990	6.72324	1.82%	6.815179	0.724332	0.092747	0.0861	0.1006	0.13613			6.263447	0.459788	0.1
13	1991	6.81518	1.67%	6.951253	0.721506	0.086143	0.0805	0.0927	0.12154			6.356209	0.45897	0.1
14	1992	6.95125	0.80%	7.022179	0.717412	0.080483	0.0756	0.0861	0.10997			6.492979	0.458274	0.1
15	1993	7.02218	0.83%	7.146157	0.715319	0.075576	0.0713	0.0805	0.10056			6.564505	0.457675	0.1
16	1994	7.14616	0.05%	7.256909	0.711726	0.071276	0.0675	0.0756	0.09275	0.7103		6.689003	0.457153	0.1
17	1995	7.25691	-0.57%	7.336841	0.708582	0.067476	0.0641	0.0713	0.08614	0.39434		6.800214	0.456696	0.1
18	1996	7.33684	-0.80%	7.38112	0.706352	0.06409	0.0611	0.0675	0.08048	0.27949		6.880551	0.45629	0.1
19	1997	7.38112	-0.59%	7.440142	0.70513	0.061055	0.0583	0.0641	0.07558	0.21892		6.92519	0.455929	0.1
20	1998	7.44014	-0.61%	7.496007	0.703515	0.058315	0.0558	0.0611	0.07128	0.18114		6.984537	0.455605	0.1
21	1999	7.49601	-0.62%	7.541404	0.702002	0.05583	0.0536	0.0583	0.06748	0.15517	0.710304	7.040694	0.455313	0.1
22	2000	7.5414	-0.52%	0	0.700784	0.053565	0.0515	0.0558	0.06409	0.13613	0.394339057	7.086356	0.455048	0.1

$$\lambda_0 = (\beta f'(x))^{1/3}$$

$$f'(x_0) = \frac{\partial y}{\partial x} = \frac{\frac{du_{(t-c)}}{dt}}{\beta\frac{du_{(t-c)}}{dt} + \frac{du_{(t)}}{dt}}$$

substituting for $f'(x)$ we get

$$\lambda_0 = \left(\beta\frac{\frac{du_{(t-c)}}{dt}}{\beta\frac{du_{(t-c)}}{dt} + \frac{du_{(t)}}{dt}}\right)^{1/3}$$

m*b	du = (4.704*0.151)*T^(0.151-1)
4.704*0.151	du_t-c = (4.704*0.151)*T_t-c^(0.
0.710304	
J '(k,k+1)= .7122*1.76 k^(.7122-1)	
1.76*.7122	
1.253472	J' coeff
-0.2878	J' exponent

Figure 25. "Entropy lambda" sheet with Lyapunov calculation.

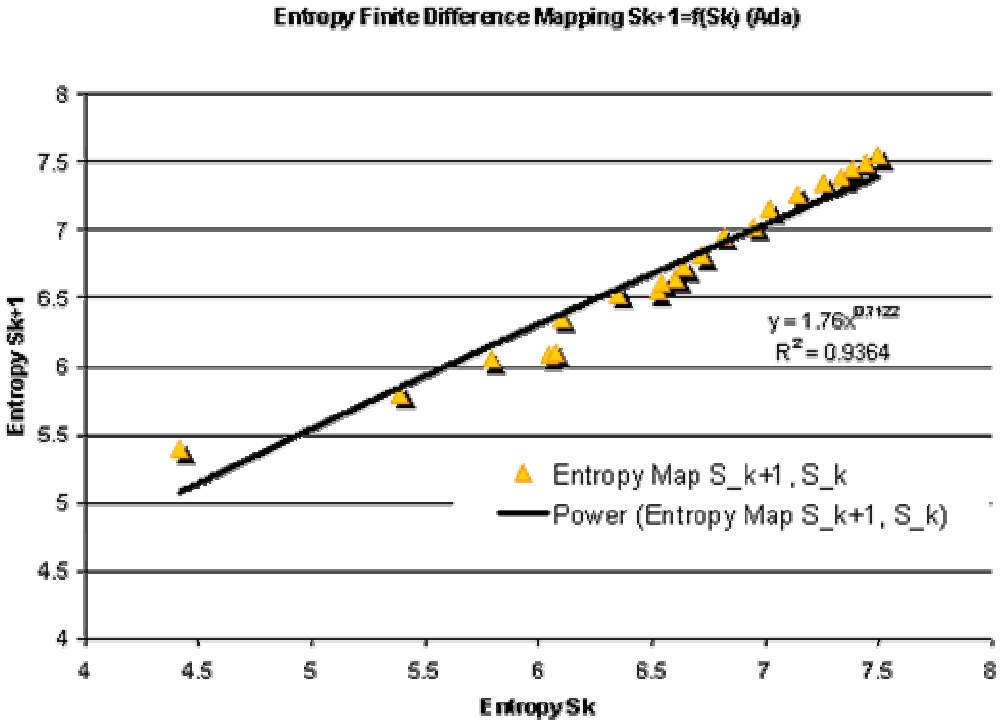


Figure 26. Map of S_{HK}, S_{HK+1} produced in fourth iteration.

3. Measure

As with iteration one, measuring user interaction with the system was unimportant. The importance of this iteration was the ability to perform *new* calculations on a standardized set of data which could then be applied to multiple datasets.

4. Learn

By fulfilling the requirements of this iteration focus switched to new data analysis methods and ways to implement greater granularity of the data presented in Interaction Three.

D. ITERATION THREE: FINALIZATION OF DATA ANALYSIS

More analysis and calculations still needed to be added. Several algorithms and procedures were added during the third iteration.

First, a higher level of granularity was needed. There were problems basing a publication's date based on the "Publication Year" field. This limited the analysis to perform computations based on yearly time steps and produced inadequate results for shorter (time-wise) datasets. The need arose to perform the calculations using monthly increments.

Secondly, there was great success with computing entropy based on each term contained in a record's descriptor field. With this success it was determined that the publication's descriptors could be combined. These combinations essentially make up the language of the dataset under scrutiny.

Saboe points out, on page 67 of his dissertation (found in Appendix C), that a publication record contains a descriptor field. A descriptor field contains a set of terms that are representative of the topics covered in that publication.

An example of the term combinations is shown below:

A publication record contains terms A, B, C. The possible term combinations are the following:

q-level = 1	q-level = 2	q-level = 3
A B C	AB BC	ABC

Q-level's are used to describe the n-tuple (single, double, triple, and so on) combination of the terms for a publication record. From the above example the language of the dataset of one publication record is {A, B, C, AB, BC, ABC}.

Third, the dataset was broken up by bands based on the production rate of an affiliation. This is described below under implement.

Finally, Saboe described a new set of calculations for describing the pressure and temperature of the dataset. The theory behind these calculations is found in his dissertation. An excerpt from the dissertation is found in Appendix C and shows that the

temperature and pressure functions are dependant on the cumulative entropy, number of terms, and number of authors.

1. Plan

The following are the set of requirements for iteration three:

- The system must perform calculations by month.
- The system must establish the vocabulary of the dataset by obtaining the combination of terms
- The system must break the dataset up into bands based on the production rate of affiliations.
- The system must compute the entropy, temperature, and pressure of the dataset on both combined and uncombined terms.

The use cases remained the same from iteration one except for two new use cases that came into existence to facilitate features of MS Access. Figure 27 shows the added use cases.

Access allowed the data to be organized better than Tech OASIS through the use of relationships and provided different data views with its query capabilities. Access was especially needed to provide the time interval relationships to the accession number field which gave the capability to perform calculations in monthly instead of yearly intervals. Access also facilitated re-usability. The same database schema was used for each dataset.

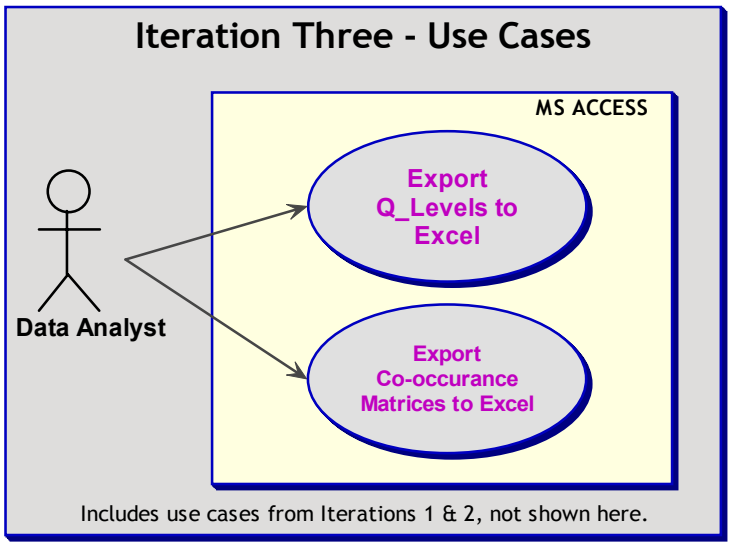


Figure 27. Iteration three use cases.

The class diagram had to be expanded to account for the use of MS Access. Shown below in Figure 28, it is apparent that MS Excel and MS Access interface with each other through the use of VBA programs.

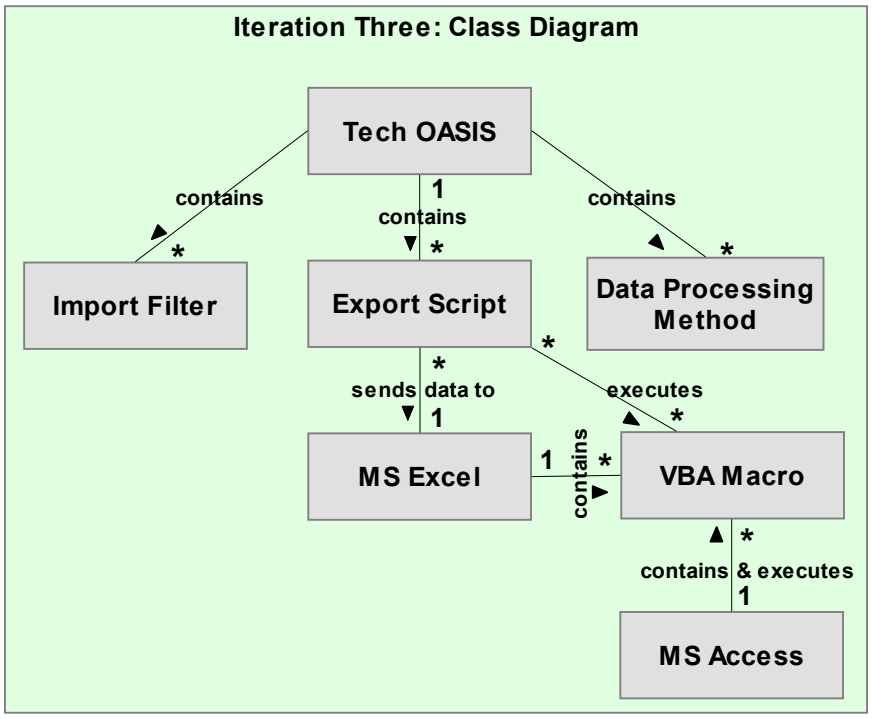


Figure 28. Iteration three class diagram.

A typical sequence of events is shown in the two sequence diagrams below.

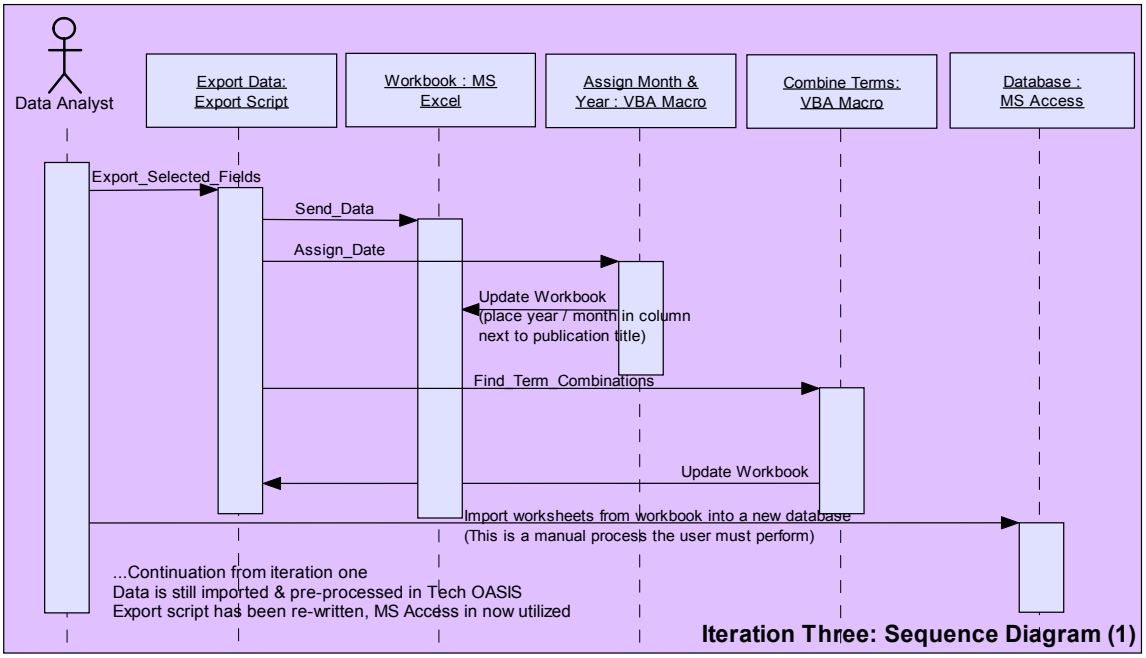


Figure 29. Iteration three sequence diagram one of two.

Figure 29 shows the following sequence of events:

1. The data is exported from Tech OASIS to a MS Excel workbook via export script.
2. The export script runs the Assign Month & Year macro.
3. The Assign Month & Year macro places the month and year the publication was entered into the INSPEC database next to the publication's title.
4. Once the Assign Month & Year macro finishes, the Combine Terms macro is called to run.
5. The workbook is updated with the combined terms.
6. The data analyst manually imports all the worksheets into a newly created MS Access database.

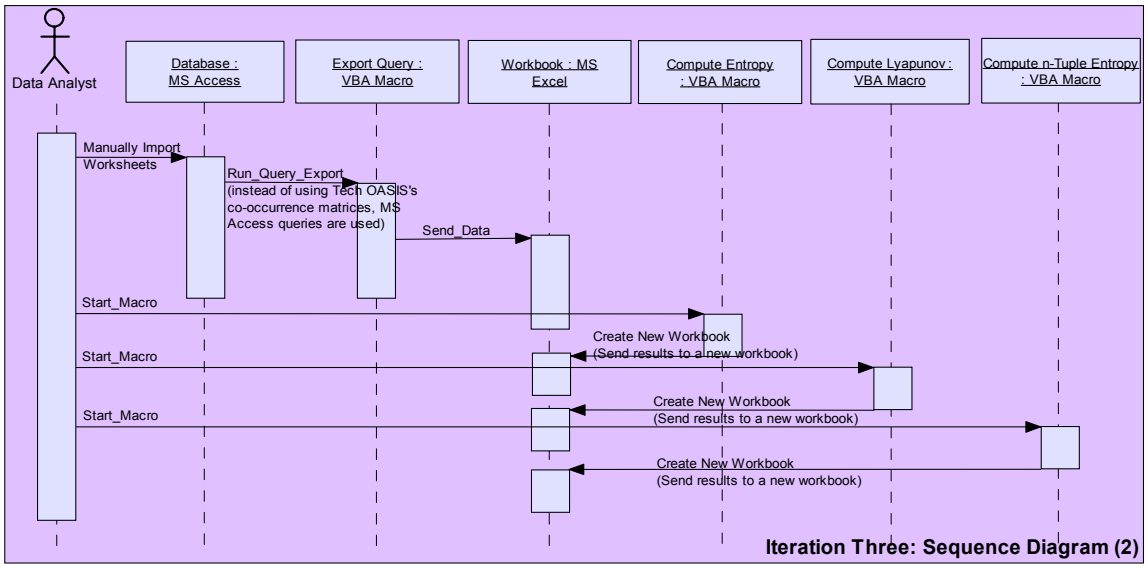


Figure 30. Iteration three sequence diagram two of two.

Figure 30 shows the following sequence of events:

1. The user manually imports the MS Excel worksheets into MS Access.
2. The user runs the Export Query macro from within MS Access.
3. The Export Query macro creates the co-occurance matrices through queries and sends the results to MS Excel.
4. The data analyst starts the compute entropy macro.
5. The Compute Entropy macro performs the entropy computations, generates the cumulative entropy summary sheet and graph, entropy lambda sheets and graphs, and temperature and pressure graphs.
6. The data analyst executes the Compute Lyapunov macro.
7. The Compute Lyapunov macro creates a new workbook and calculates the lyapunov exponent.
8. The data analyst runs the Compute n-Tuple Entropy macro.
9. The Compute n-Tuple Entropy macro creates a new workbook where the results are stored.

As part of the database design an entity relation diagram (ERD) was created to show the relationships between the export data fields from Tech OASIS. Figure 31 shows the ERD.

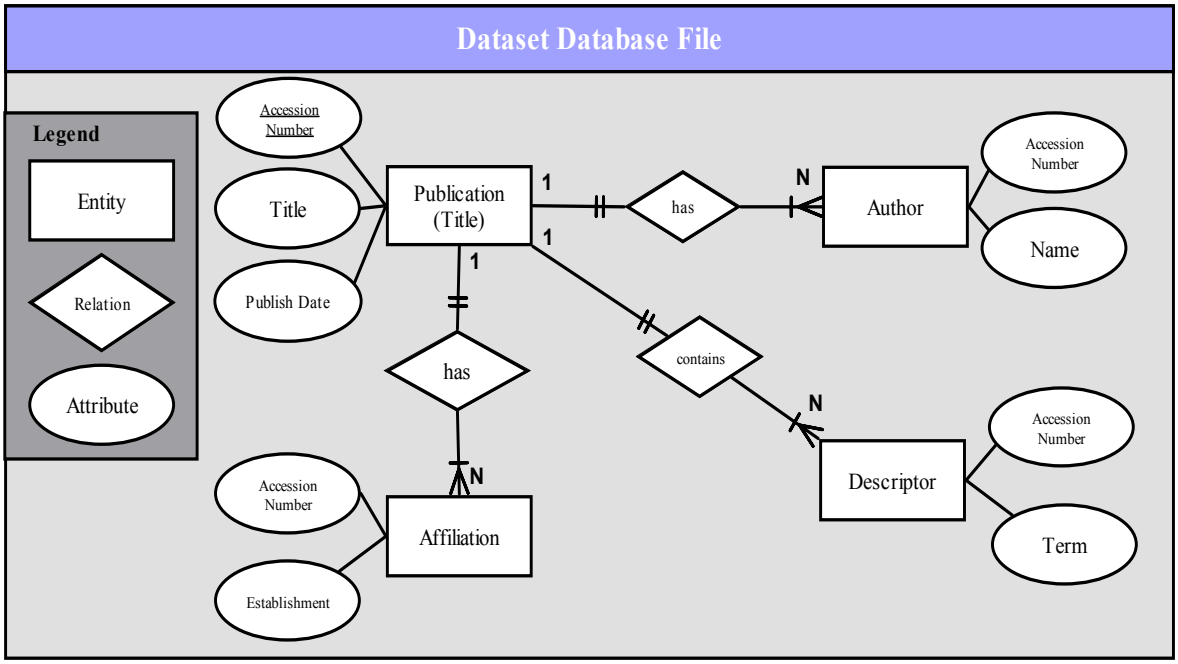


Figure 31. Iteration three database entity relation diagram.

2. Implement

a. Calculations by Month

The export script was re-written. Instead of exporting a co-occurance matrix created by Tech OASIS, the export script now sends a list of data from a user selected data field associated with accession numbers to maintain record divisions and relationships.

During execution of the Tech OASIS export script, the user is prompted to select the data fields to export. The first prompt, shown in Figure 32, asks the user to select a unique identifier field. The next series of prompts ask the user to select the data fields, shown in Figure 33. A typical list of data fields is shown in Figure 34. The INSPEC accession number field is the one usually chosen as the unique identifier. The other important fields for this research included the *Title*, *Affiliation*, *Authors*, and *Descriptors* fields.

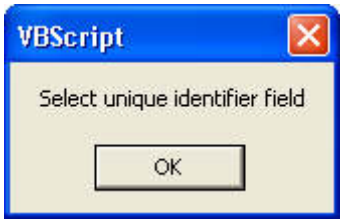


Figure 32. Prompt for unique identifier.

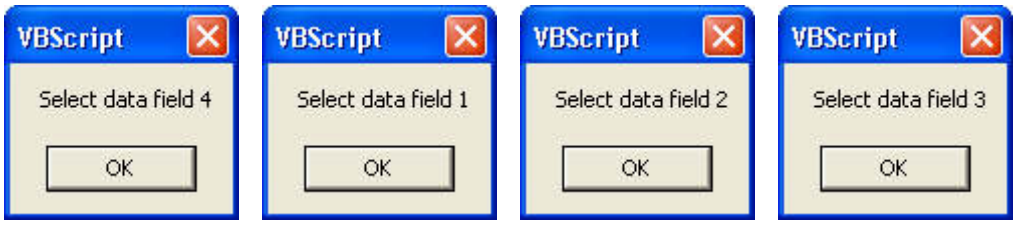


Figure 33. Prompts to select data fields to export.

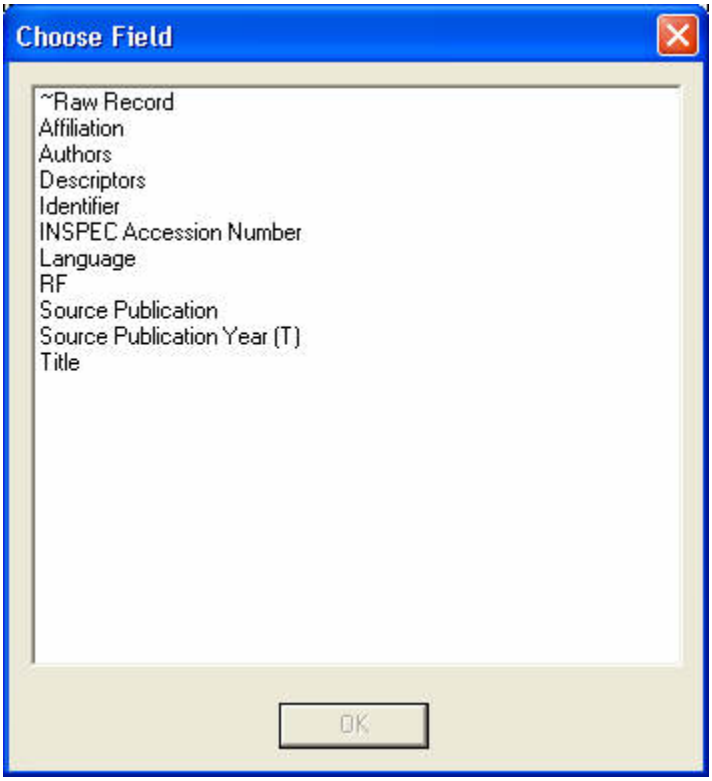


Figure 34. List of fields to select.

Once the fields are selected, the export script creates a co-occurrence matrix in Tech OASIS of the data field values on the y-axis and the unique identifier field on the x-axis.

		Title	1	2	3	4	5	6	7	8	9	10	11	12	13
		# Records	1	1	1	1	1	1	1	1	1	1	1	1	1
Accession N	# Records	1	02046766	02076615	02078644	02106610	02143179	02159589	02186146	02186147	02186188	02186189	02186190	02187402	02187403
1	2	The MCC Leonardo Proje													
2	2	Early field experience wit													
3	2	A comparison of design			1										
4	2	Technology transfer													
5	1	Software engineering: str					1								
6	1	The transfer of university						1							
7	1	Imperial Software Techno							1						
8	1	Technology transfer take								1					
9	1	Training for software eng									1				
10	1	The 'Chinese Lunch' synd										1			
11	1	The role of professional d											1		

Figure 35. Export script co-occurrence matrix.

The export script traverses this co-occurance matrix (shown in Figure 35) in a 2-d path. A column is selected and each row is examined in that column. If the value of that column and row combination is not empty then the value in the column header is sent along with the value in the row header to MS Excel. Figure 36 shows the resulting values in MS Excel. The final source code for the export script is found in Appendix K.

Book1

	A	B	C	D	E
1	INSPEC A	Title			
2	2046766	Technology transfer of a software engine			
3	2076615	Early experiences regarding SDMS intro			
4	2078644	A comparison of design strategies for so			
5	2106610	Technology transfer: the key to software			
6	2143179	Software engineering: strategies for tech			
7	2159589	The transfer of university software for ind			
8	2186146	Imperial Software Technology Limited			

Title / Sheet1 / Shee

Figure 36. Data exported for the Title field into MS Excel.

After the exportation completed, the month and year needed to be assigned to the publications. This was accomplished by another VBA macro. The macro traversed the worksheet containing the title of the publications while examining a list of accession numbers to see what month and year to place next to the title. See Appendix L for the list of accession numbers and the source code of the macro to assign the dates. Figure 37 shows the results of the title worksheet after macro completed.

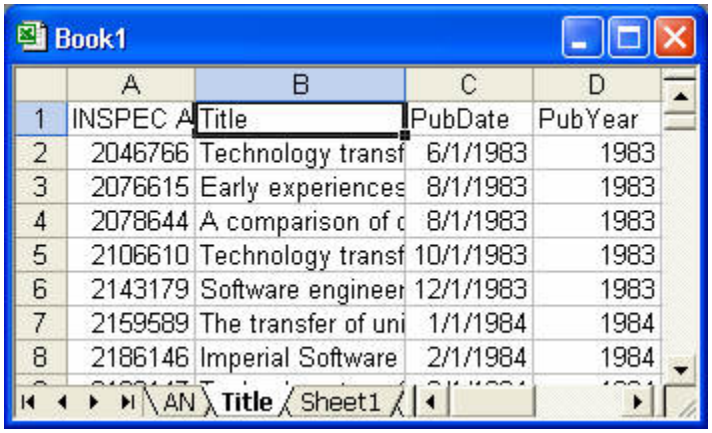


Figure 37. Exported data after the date have been assigned to the Title field.

b. Term Combination

After applying the date to the records the descriptors were ready to be combined to generate the language of the dataset. The descriptors needed to be combined in a way that the powerset of the descriptors field is represented for every publication. Another macro named the "Term Combination" macro was written to accomplish this. The source code for the macro is found in Appendix M.

The figures below show the results of the term combination macro.

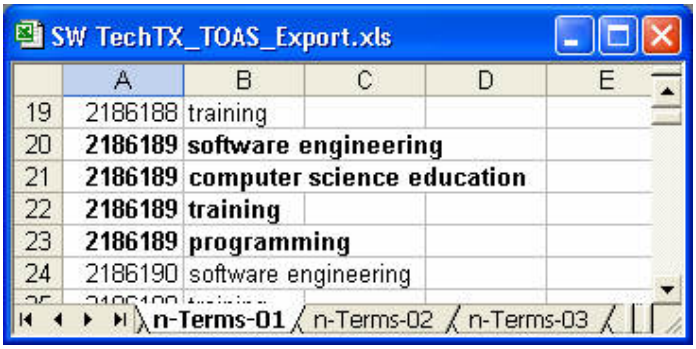


Figure 38. Single descriptors.

Figure 38 shows the single descriptors of the publication indexed with the accession number 2186189 in bold. The worksheet of single descriptors is used as input to the term combination macro. Figure 39 shows the terms of the same records combined to create double terms, Figure 40 shows those same terms combined to create the triple terms, and Figure 41 shows them combined to create the quadruple terms. Thus the powerset of the descriptor field for that one record is complete.

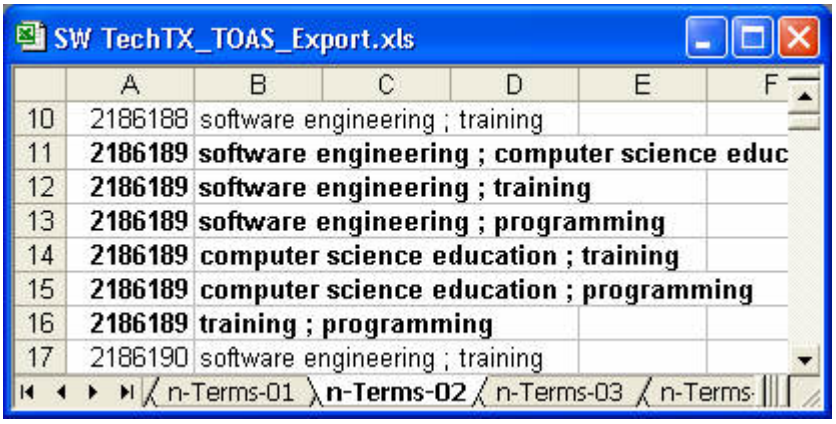


Figure 39. Terms combined to make double terms.

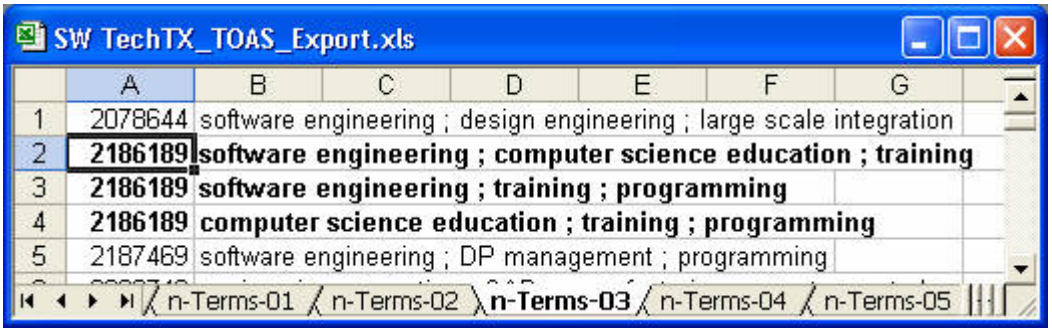


Figure 40. Terms combined to make triple terms.

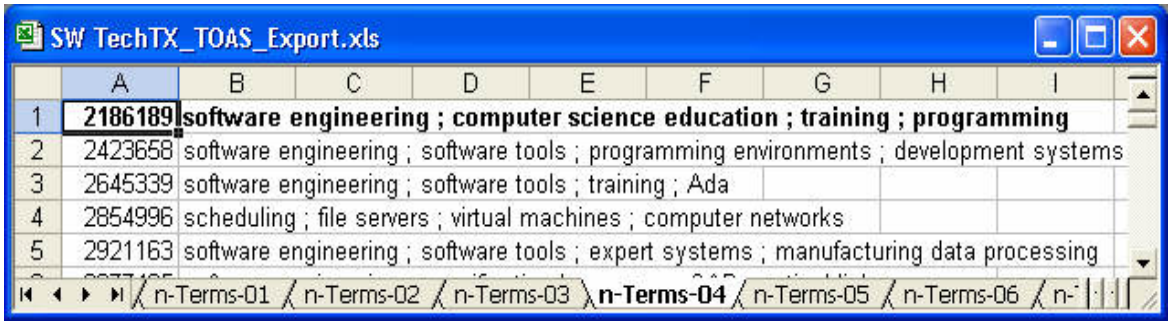


Figure 41. Terms combined to make quadruple terms.

c. Import into MS Access

Next all the sheets created from the Tech OASIS exportation script and the Term Combination macro need to be imported into Microsoft Access.

The relationships outlined in the above ERD make the database useful. The relationships are exploited through the use of queries.

The process takes many tedious steps to complete for each dataset Which will have to be automated in a future iteration. The process of creating and population the database is as follows:

1. Create a new database
2. Select the import command under the file menu and under "Get External Data" see Figure 42 below.
3. Select the filename

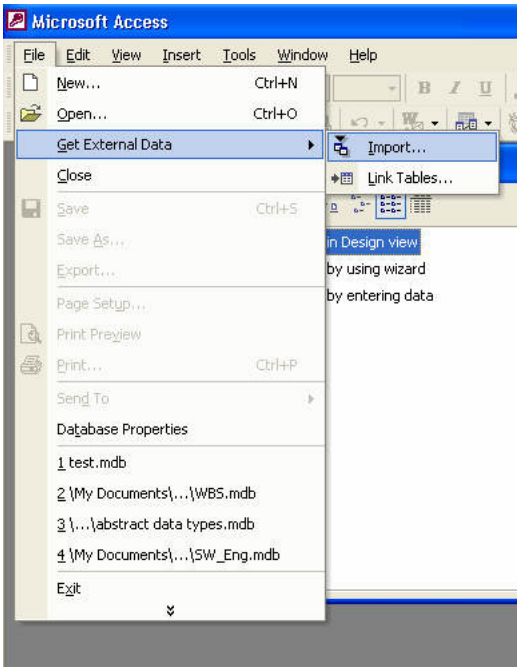


Figure 42. MS Access import command.

4. Then select the worksheet to be imported. Shown in Figure 43.
5. Set the column heading and primary keys. As shown on the ERD in Figure 31, the only primary key needed is for the "Title" table.

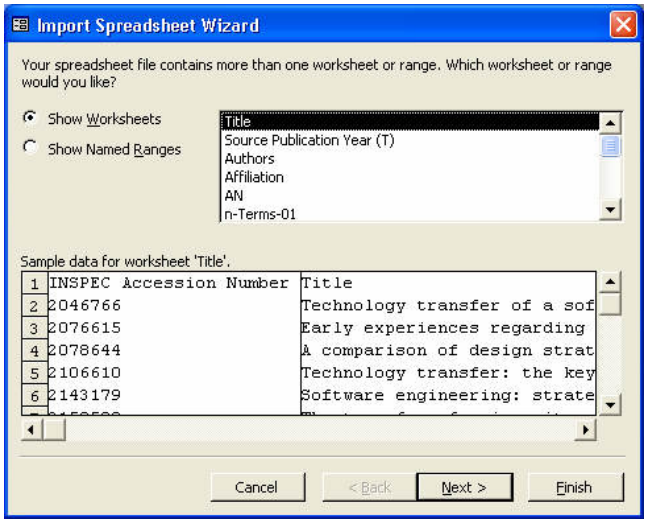


Figure 43. Worksheet selection.

6. The queries need to be created after the importation of the worksheets into MS Access. The MS Access query designs are show below in Figures 44-48.

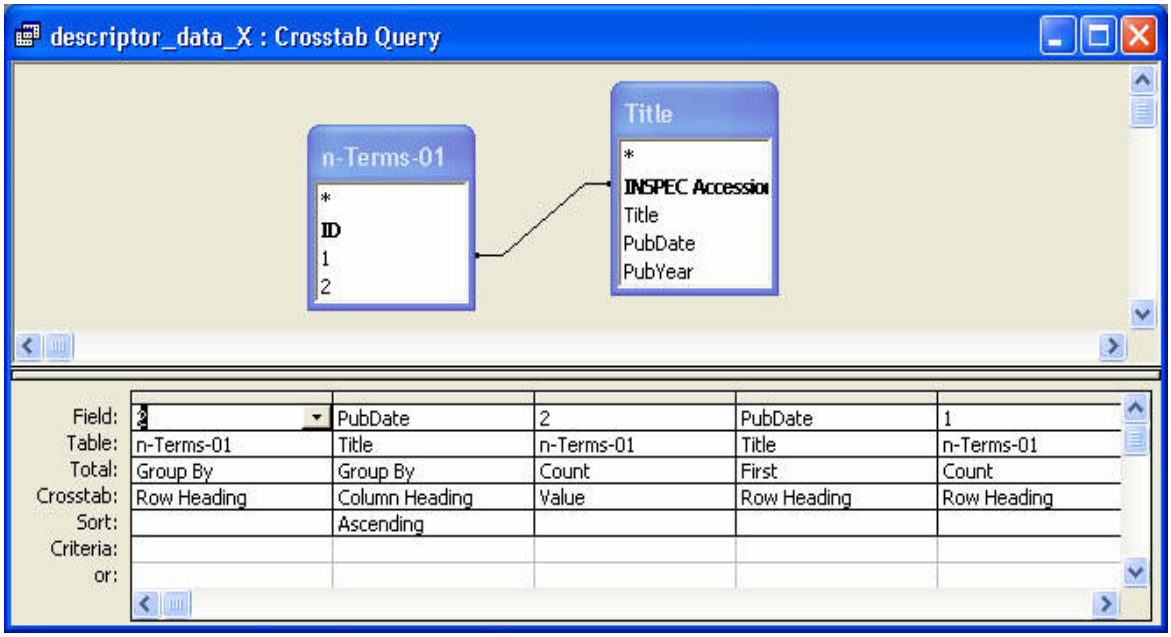


Figure 44. Co-occurrence matrix query design (descriptor instances over time).

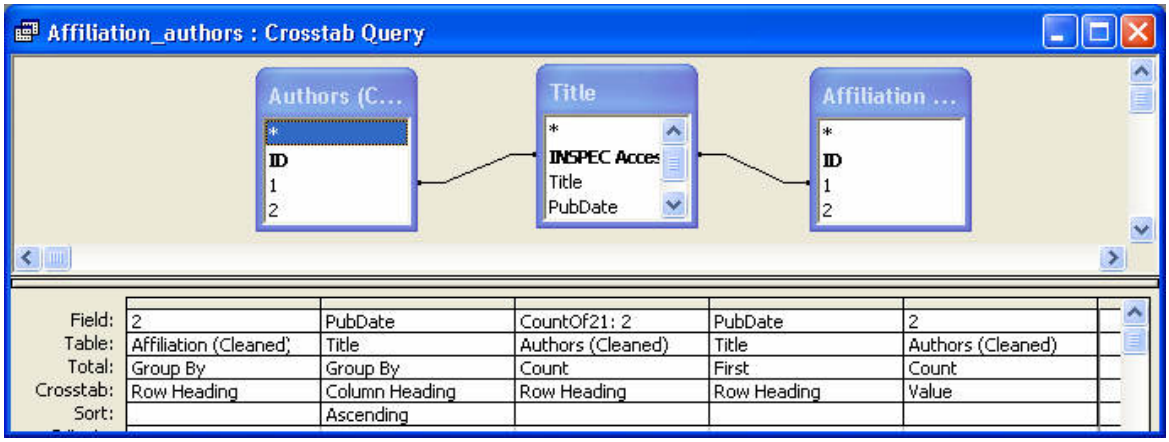


Figure 45. Co-occurrence matrix query (authors instances per affiliation over time)

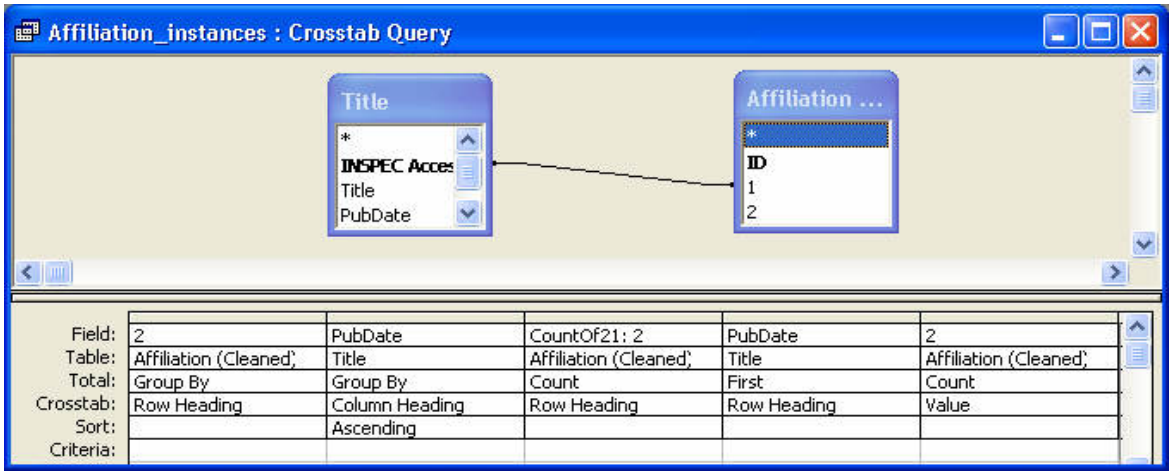


Figure 46. Co-occurrence matrix query design (affiliation instances per affiliation over time)

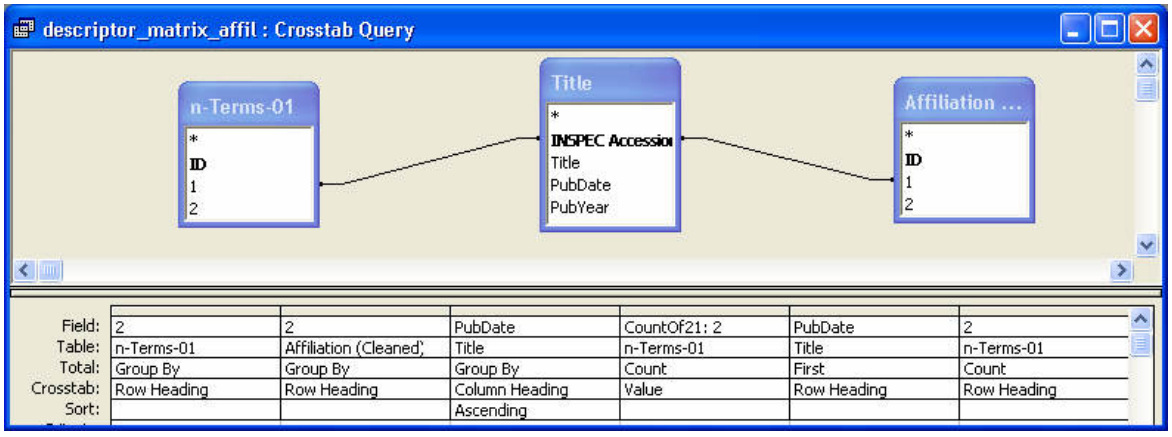


Figure 47. Co-occurrence matrix query design (descriptor instances over time – descriptor and affiliation pair).

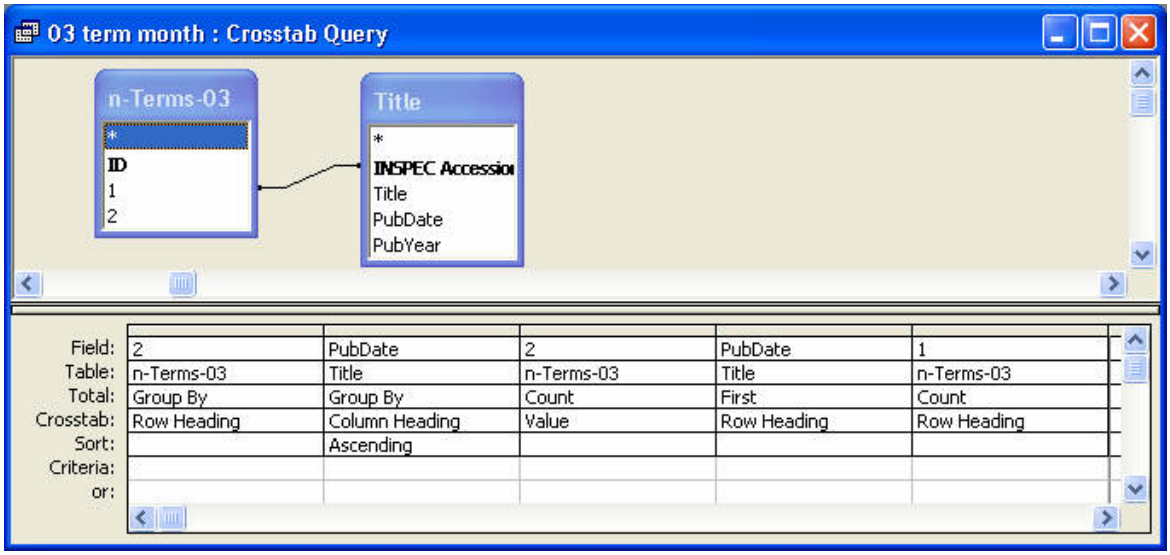


Figure 48. Co-occurrence matrix query design (n-term instances over time)

The last query shown must be replicated for all the q-levels or term combination tables.

The queries' results then need to be exported to allow the analysis macros to run on them. This is done by selected the export command under the file menu while viewing the query. Another VBA macro was created to speed up the process of exporting the queries relating to term combination. The source code to the macro is shown below in Figure 49.

```
Sub OutputDataToExcel()

For i = 1 To 8  '1 to the number of q_levels, this must be changed for each dataset.
   If i < 10 Then
   'terms:
      DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "0" & i & " term month",
                "c:\dataout\WBS_qlevels_term_month.xls"
   Else
   'terms
      DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "" & i & " term month",
                "c:\dataout\WBS_qlevels_term_month.xls"
   End If
Next i

End Sub
```

Figure 49. Code to export queries to MS Excel

d. Affiliation Distribution Analysis Macro

With the co-occurrence matrices in MS Excel, powerful VBA macros could be ran on the data. The VBA macros reside in MS Excel workbooks along with skeletons of the charts. The macros copy the charts into the workbook after the analysis results are stored in the workbook.

The first macro is called "Affiliation Distribution". This distributes the records based on the production of each affiliation. This macro requires four co-occurrence matrices exported from MS Access. The four needed are the following:

- Matrix of descriptor (term) instances per descriptor over time. See query in Figure 44.
- Matrix of author instances per affiliation over time. See query in Figure 45.
- Matrix of affiliations instances per affiliation over time. See query in Figure 46.
- Matrix of descriptor instances per descriptor and affiliation pair over time. See query in Figure 47.

The records are distributed into four bands based the number of records produced by an affiliation. Figure 50 shows the created distribution sheet.

Statistics						
Mean	1.892197		Band Range			
Stdev	2.522752		Band_D	Band_C	Band_B	Band_A
Sum	4195	from	0	4.415	6.938	9.46
Count	2217	to	4.415	6.938	9.46	
Bin	Frequency					
1	1566					
2	312	2217 total affiliations that published 4195 records				
3	121	52 records in Band A				
4	62	39 affiliation in Band B				
6.938	65	65 affiliations in Band C				
9.46	39	2061 affiliations in Band D				
15	38					
20	9					
25	4					
> 25	1					

Figure 50. Distribution of affiliations into bands for the Ada dataset.

The bands are determined by using the mean (μ) and the standard deviation (σ) of the number of publications each affiliation produced. The affiliations in each band are based on the number of records produced n and are determined as follows:

Band D: $0 > n < \mu+1\sigma$.
Band C: $\mu+1\sigma > n \leq \mu+2\sigma$.
Band B: $\mu+2\sigma > n \leq \mu+3\sigma$.
Band A $\mu+3\sigma > n$

The discussion of the purpose of dividing the dataset into bands is explained in Saboe's dissertation on pages 126-132 which have been included in Appendix C.

After the affiliations have been distributed, entropy, temperature, and pressure can be calculated for each band and the world (no affiliation separation). The results from the bands provided gave poor results so the focus of this research centered on the dataset as a whole. After the calculation algorithms have finished, summary sheets and charts are generated such as the one show in Figure 51.

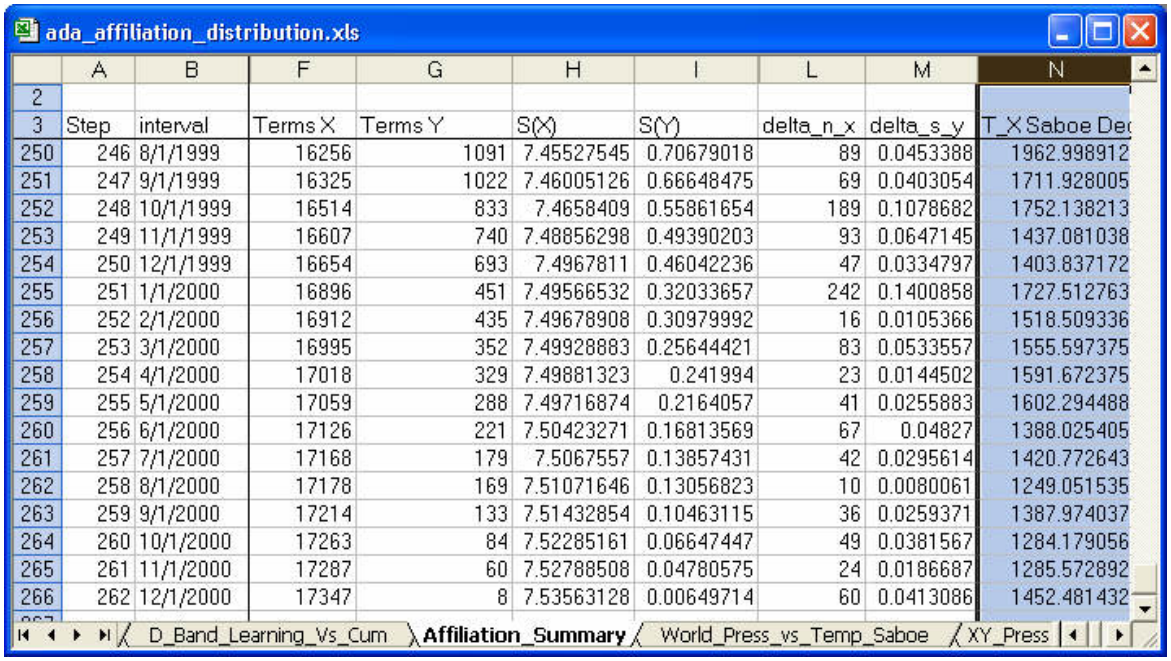
ada_affiliation_distribution.xls

	A	B	C	D	E	F	G	H	I
1									
2									
3	Step	interval	A_Band Instanc	B_Band Instanc	C_Band Instanc	D_Band Entrop	Sum Band Insta	World Entropy I	Diff World & Ban
207	204	10/1/1999	3130	1171	1483	10730	16514	16514	0
208	205	11/1/1999	3173	1171	1483	10780	16607	16607	0
209	206	12/1/1999	3173	1171	1483	10827	16654	16654	0
210	207	1/1/2000	3217	1197	1501	10981	16896	16896	0
211	208	2/1/2000	3217	1197	1506	10992	16912	16912	0
212	209	3/1/2000	3221	1197	1517	11060	16995	16995	0
213	210	4/1/2000	3234	1197	1520	11067	17018	17018	0
214	211	5/1/2000	3236	1203	1520	11100	17059	17059	0
215	212	6/1/2000	3258	1203	1523	11142	17126	17126	0
216	213	7/1/2000	3274	1203	1528	11163	17168	17168	0
217	214	8/1/2000	3274	1209	1528	11167	17178	17178	0
218	215	9/1/2000	3284	1209	1528	11193	17214	17214	0
219	216	10/1/2000	3300	1209	1533	11221	17263	17263	0
220	217	11/1/2000	3304	1209	1537	11237	17287	17287	0
221	218	12/1/2000	3304	1209	1542	11292	17347	17347	0
222									

Affiliation_authors | descriptor_data_X | **Instances Summary** | World_Cumulative_Entropy_X | descriptor_data_

Figure 51. Summary sheet showing descriptor term instances for the Ada dataset.

Temperature of a time step is calculated from the change in cumulative entropy and change in the number of terms in the dataset. Figure 52 shows the results of the temperature calculation highlighted on a summary sheet. Saboe describes this calculation on pages 116-121 of his dissertation which have been included in Appendix C.



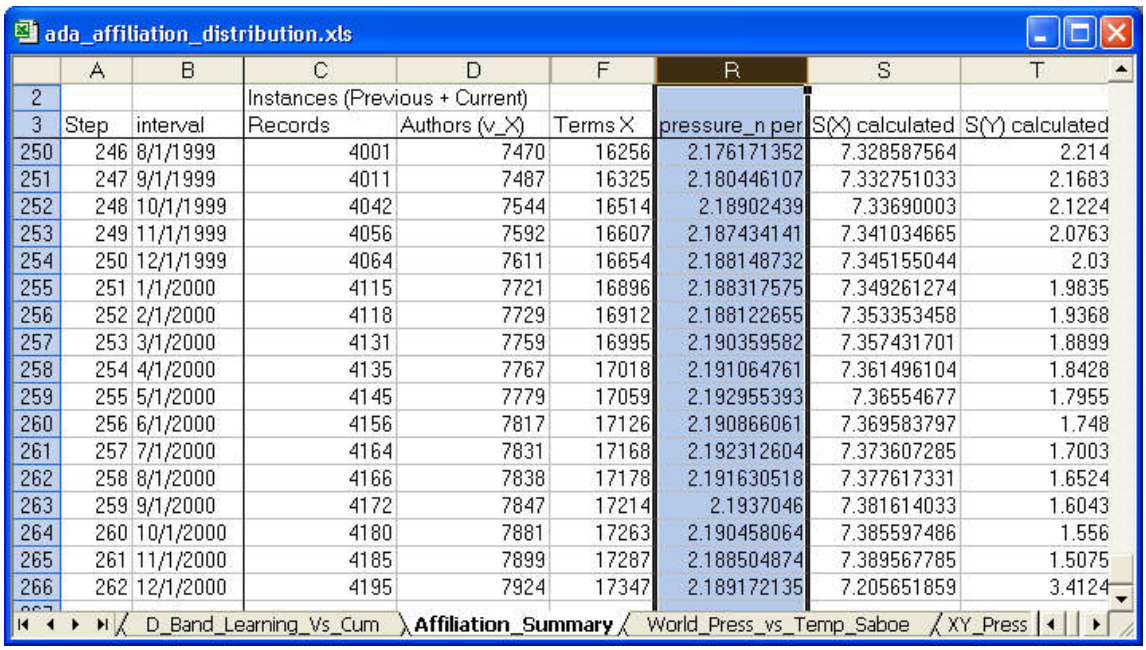
ada_affiliation_distribution.xls

	A	B	F	G	H	I	L	M	N
2									
3	Step	interval	Terms X	Terms Y	S(X)	S(Y)	delta_n_x	delta_s_y	T_X Saboe Dec
250	246	8/1/1999	16256	1091	7.45527545	0.70679018	89	0.0453388	1962.998912
251	247	9/1/1999	16325	1022	7.46005126	0.66648475	69	0.0403054	1711.928005
252	248	10/1/1999	16514	833	7.4658409	0.55861654	189	0.1078682	1752.138213
253	249	11/1/1999	16607	740	7.48856298	0.49390203	93	0.0647145	1437.081038
254	250	12/1/1999	16654	693	7.4967811	0.46042236	47	0.0334797	1403.837172
255	251	1/1/2000	16896	451	7.49566532	0.32033657	242	0.1400858	1727.512763
256	252	2/1/2000	16912	435	7.49678908	0.30979992	16	0.0105366	1518.509336
257	253	3/1/2000	16995	352	7.49928883	0.25644421	83	0.0533557	1555.597375
258	254	4/1/2000	17018	329	7.49881323	0.241994	23	0.0144502	1591.672375
259	255	5/1/2000	17059	288	7.49716874	0.2164057	41	0.0255883	1602.294488
260	256	6/1/2000	17126	221	7.50423271	0.16813569	67	0.04827	1388.025405
261	257	7/1/2000	17168	179	7.5067557	0.13857431	42	0.0295614	1420.772643
262	258	8/1/2000	17178	169	7.51071646	0.13056823	10	0.0080061	1249.051535
263	259	9/1/2000	17214	133	7.51432854	0.10463115	36	0.0259371	1387.974037
264	260	10/1/2000	17263	84	7.52285161	0.06647447	49	0.0381567	1284.179056
265	261	11/1/2000	17287	60	7.52788508	0.04780575	24	0.0186687	1285.572892
266	262	12/1/2000	17347	8	7.53563128	0.00649714	60	0.0413086	1452.481432

D_Band_Learning_Vs_Cum | Affiliation_Summary | World_Press_vs_Temp_Saboe | XY_Press

Figure 52. Summary sheet showing temperature calculation for the Ada dataset.

Saboe describes pressure as the calculation of messages processed per node. Figure 53 shows the results of the pressure calculation highlighted on a summary sheet.



ada_affiliation_distribution.xls

	A	B	C	D	F	R	S	T
2			Instances (Previous + Current)					
3	Step	interval	Records	Authors (v_X)	Terms X	pressure_n per	S(X) calculated	S(Y) calculated
250	246	8/1/1999	4001	7470	16256	2.176171352	7.328587564	2.214
251	247	9/1/1999	4011	7487	16325	2.180446107	7.332751033	2.1683
252	248	10/1/1999	4042	7544	16514	2.18902439	7.33690003	2.1224
253	249	11/1/1999	4056	7592	16607	2.187434141	7.341034665	2.0763
254	250	12/1/1999	4064	7611	16654	2.188148732	7.345155044	2.03
255	251	1/1/2000	4115	7721	16896	2.188317575	7.349261274	1.9835
256	252	2/1/2000	4118	7729	16912	2.188122655	7.353353458	1.9368
257	253	3/1/2000	4131	7759	16995	2.190359582	7.357431701	1.8899
258	254	4/1/2000	4135	7767	17018	2.191064761	7.361496104	1.8428
259	255	5/1/2000	4145	7779	17059	2.192955393	7.36554677	1.7955
260	256	6/1/2000	4156	7817	17126	2.190866061	7.369583797	1.748
261	257	7/1/2000	4164	7831	17168	2.192312604	7.373607285	1.7003
262	258	8/1/2000	4166	7838	17178	2.191630518	7.377617331	1.6524
263	259	9/1/2000	4172	7847	17214	2.1937046	7.381614033	1.6043
264	260	10/1/2000	4180	7881	17263	2.190458064	7.385597486	1.556
265	261	11/1/2000	4185	7899	17287	2.188504874	7.389567785	1.5075
266	262	12/1/2000	4195	7924	17347	2.189172135	7.205651859	3.4124

D_Band_Learning_Vs_Cum | Affiliation_Summary | World_Press_vs_Temp_Saboe | XY_Press

Figure 53. Summary sheet showing pressure calculation for the Ada dataset.

Figure 54 shows the resultant graph from the two calculations of pressure and temperature that was used in Dr. Saboe's dissertation on page 120.

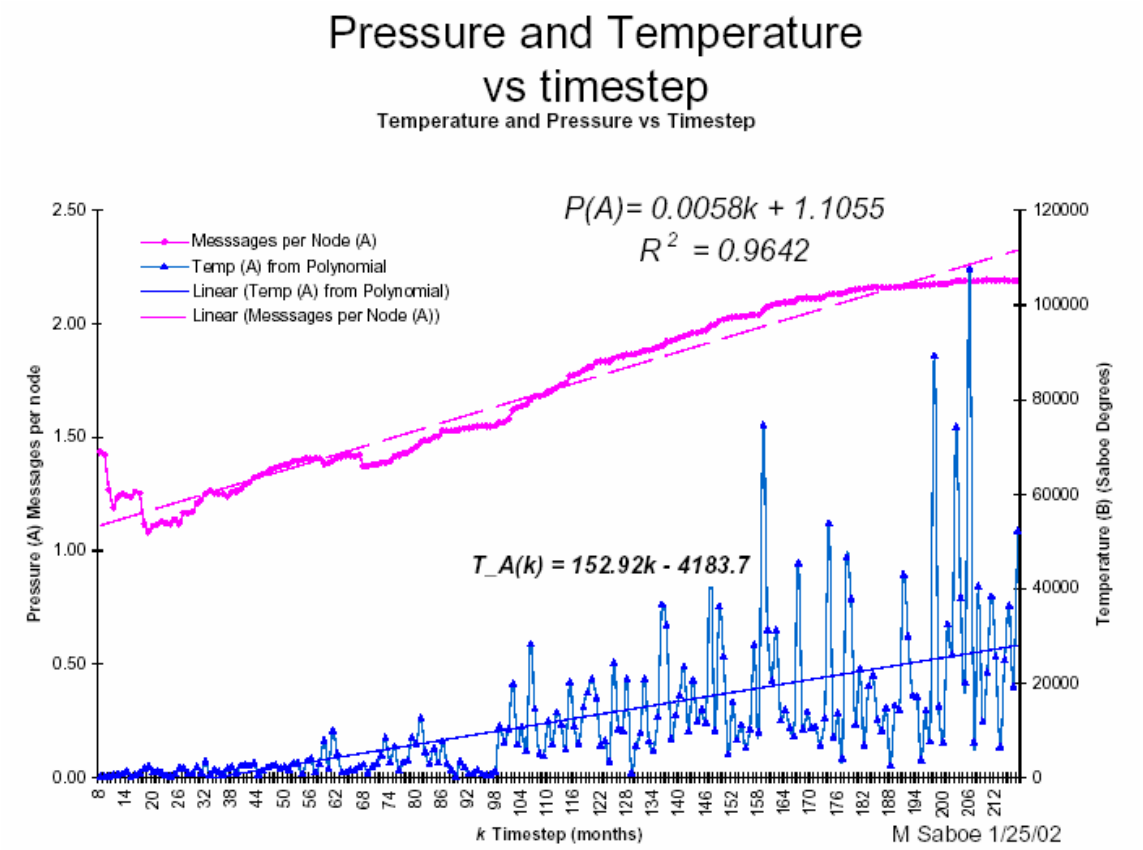


Figure 54. Pressure and Temperature Graph

Several other analyses are produced from the "Affiliation Distribution" macro, but are not significant enough to report here. See Appendix N for the final source code and Appendix H for all the output charts.

e. Entropy Lambda Analysis Macro

The second macro is called "Entropy Lambda". This macro uses results from the first macro, specifically the "Affiliation Summary" sheet shown in Figures 52 and 53. During this iteration the macro is refined to provide more precise results. Instead of guessing the beta percentage (see Saboe 161-162, 201-202 in Appendix C) the beta

percentage is determined. Source code is found in Appendix O. The results of this macro are shown below in Figures 55 and 56.

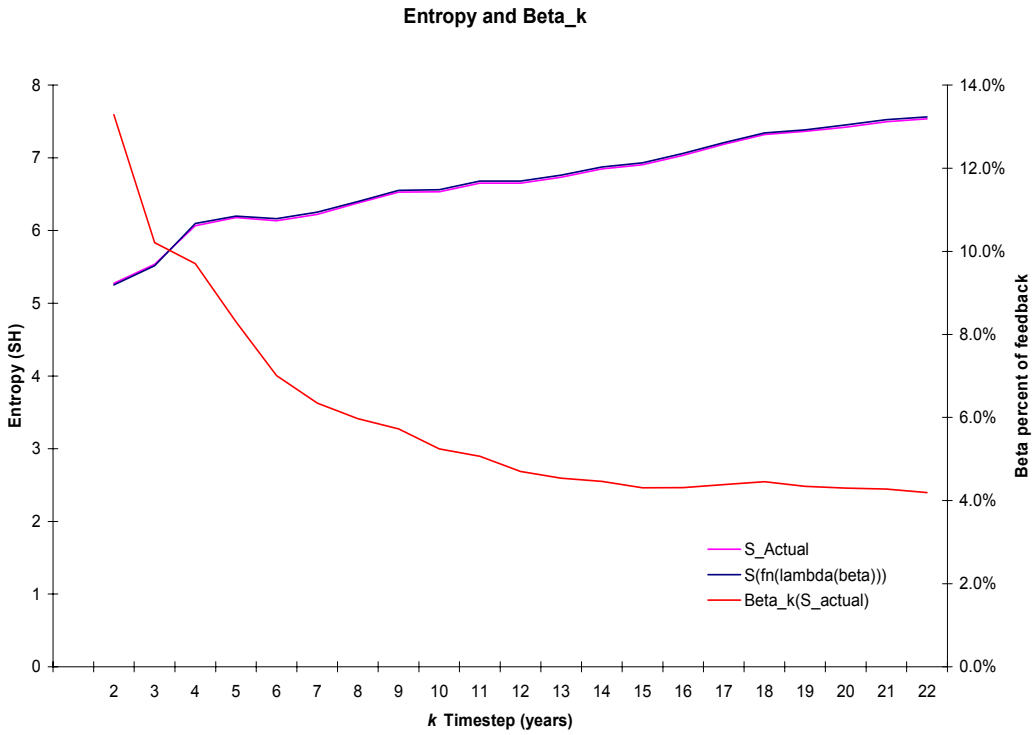


Figure 55. Entropy over time with beta fit.

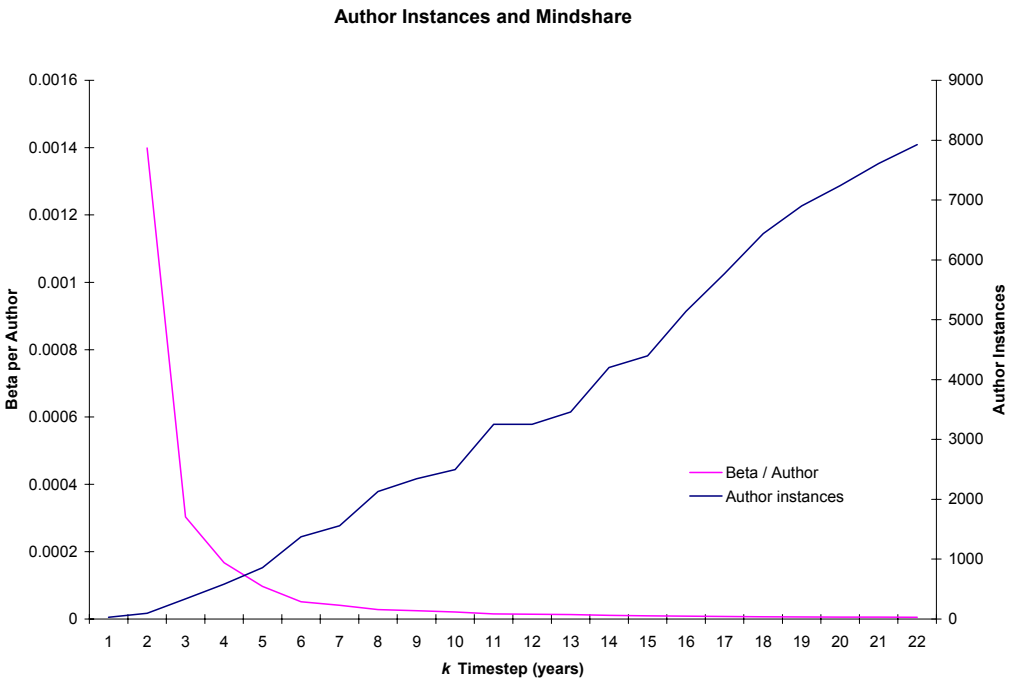


Figure 56. Author instances over time.

f. Q Level Analysis Macro

The final analysis macro is called "Q_Level_Analysis". This macro analyzes the term combinations. The inputs to this macro are the co-occurrence matrices of the q-levels. The matrices come from the query design shown in Figure 48. The query must be ran on every term combination table.

From the results it can be seen the "Q Level Analysis" macro determines the following.

- Figures 57 & 58 - Number of term instances in all the q-levels over time.
- Figure 59 – Cumulative entropy of the terms in all the q-levels over time.
- Figure 60 – Temperature of the dataset base on the term combinations.

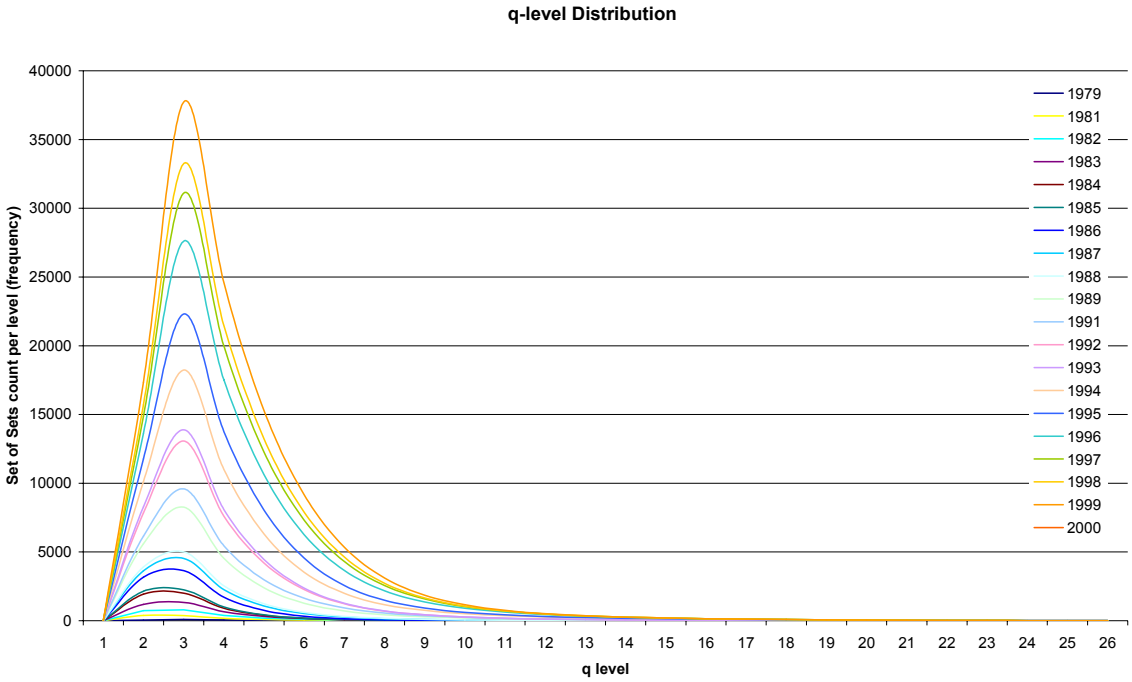


Figure 57. Term combination distribution.

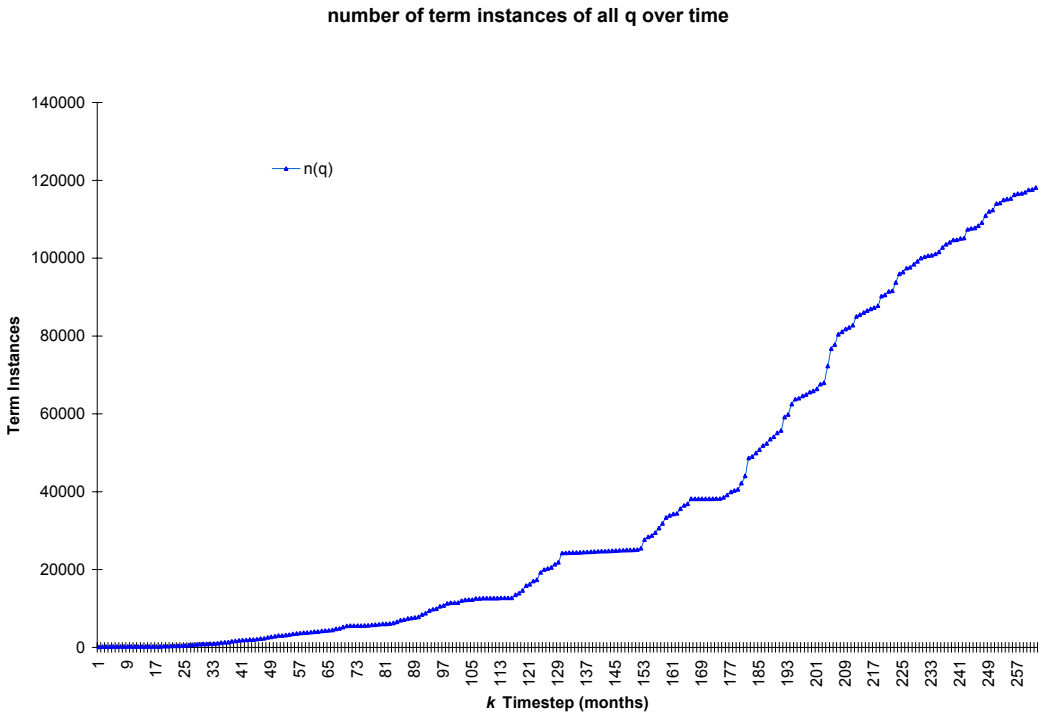


Figure 58. Number of combined-term instances.

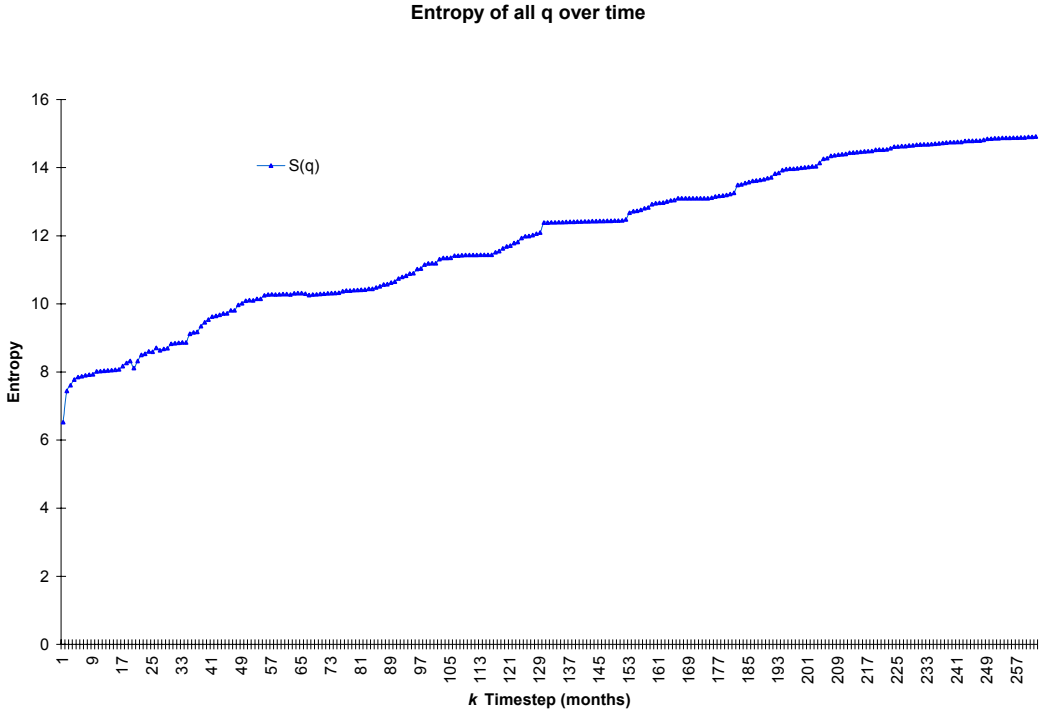


Figure 59. Entropy of combined-terms.

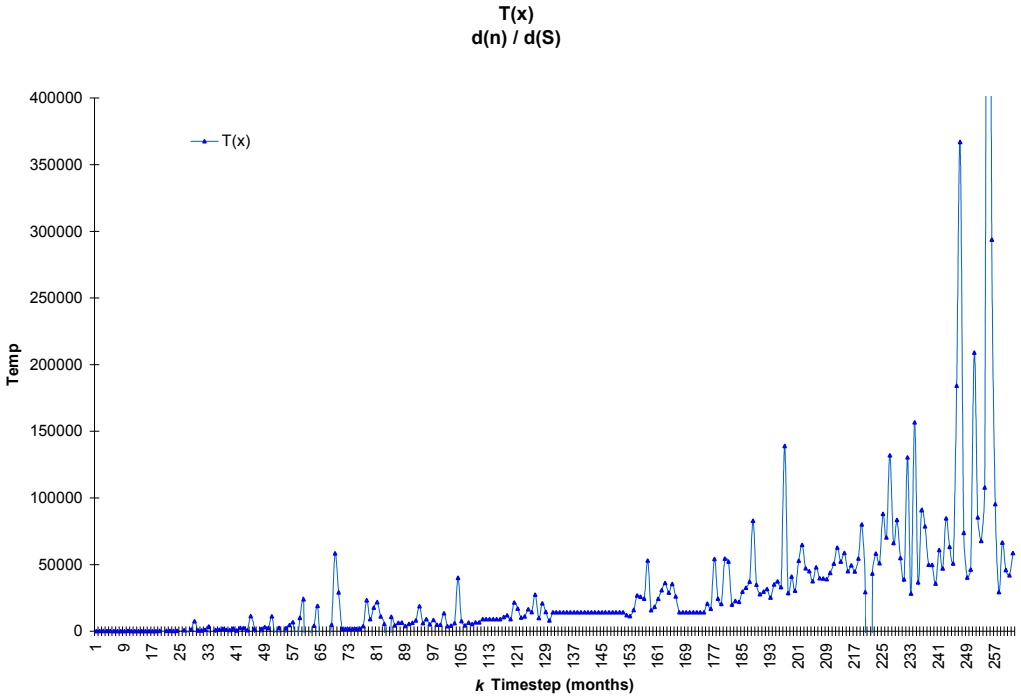


Figure 60. Temperature of the dataset.

All the output charts are shown in Appendix J. The code for the "Q Level Analysis" macro is presented in Appendix P.

3. Measure

With the addition of new analysis techniques and the refinement of old ones, more accurate and precise analysis was realized during this iteration. Dr. Michael Saboe was able to complete his dissertation and obtain his PhD.

Measured user interaction for this iteration found a major problem. The main user is the system developer and he is the only one who knows how to use these analysis functions due to the substantial amount of data preparation required to allow the functions to execute correctly.

4. Learn

Most of the data analysis techniques finalized in this iteration are ready for production. One analysis area that needs to be looked at is with the banding. The banding analysis charts from the "Affiliation Distribution" macro was not used and need to be revised or omitted from future iterations.

The major lesson learned from this iteration is that in order for the above analysis techniques to be applicable to future research the amount of user-interaction needs to be greatly reduced.

E. ITERATION FOUR: PACKAGING & INTEGRATION

With Iteration three marking the final development stage of the data analysis process, the next logical step is to develop an application to manage the process. Iteration four looks at taking the previously developed analysis techniques and packing them into one application. This new application has been dubbed DataThermometer. DataThermometer consists of a graphical user interface (GUI) that minimizes the amount of user interaction by providing one interface to the data analysis functions and the results. The Major features of DataThermometer are listed in Table 3.

Feature #	Feature
1.	Imports data from Microsoft Excel spreadsheets.
2.	Stores the data in a relational database.
3.	All functionality is performed by the user through DataThermometer's graphical user interface (GUI).
4.	User-selected data processing methods are applied to the data.
5.	The results of the data analysis are in the form of Microsoft Excel spreadsheets.
6.	DataThermometer keeps track of the data processing methods already ran on a dataset.

Table 3. Major Features of DataThermometer.

Implementation for iteration four was not finished due to the death of Dr. Saboe. Even though implementation never occurred, extensive planning was still accomplished. The next section presents the class, use case, and sequence diagrams. The Software Requirements Specification (SRS) document, found in Appendix D, shows a detailed

overview of the project along with and the use cases descriptions presented in Appendix E.

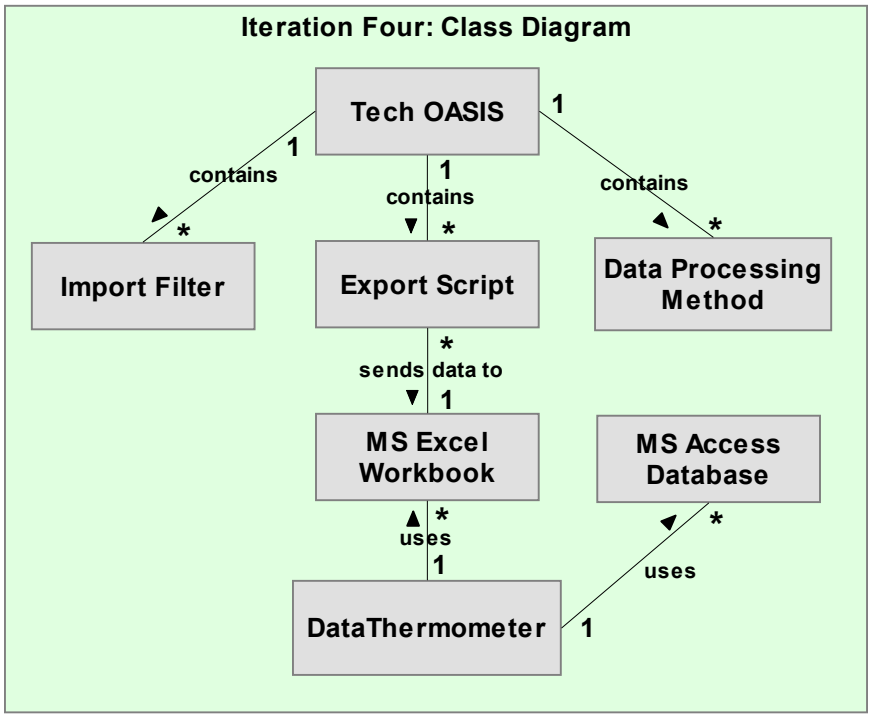


Figure 61. Iteration four class diagram.

The above class diagram shows that DataThermometer links many MS Excel workbooks to many MS Access databases. The workbooks hold the data to be imported, and the results of the analysis functions. The databases contain the imported data and queries to re-organize and manipulate the data.

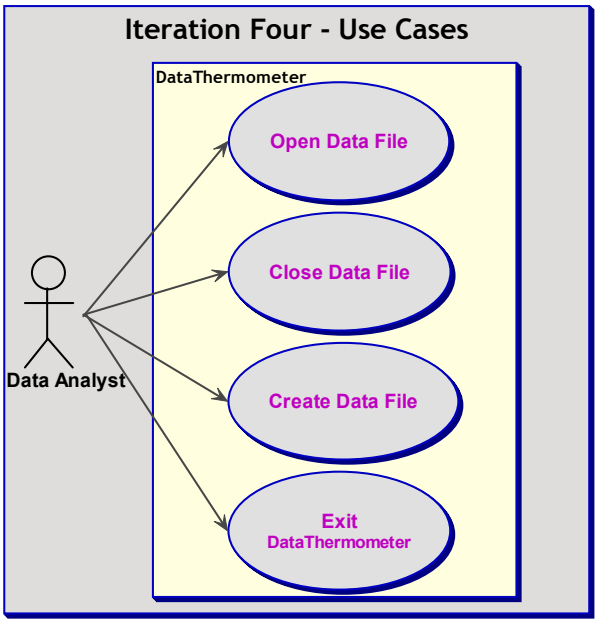


Figure 62. Iteration four use cases.

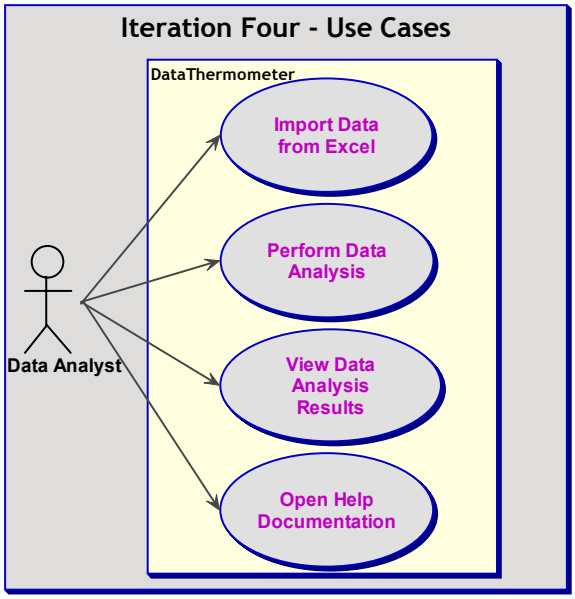


Figure 63. Iteration four more use cases.

The above use case diagrams show the functions DataThermometer provides to its users. Each use case is described in detail in Appendix E.

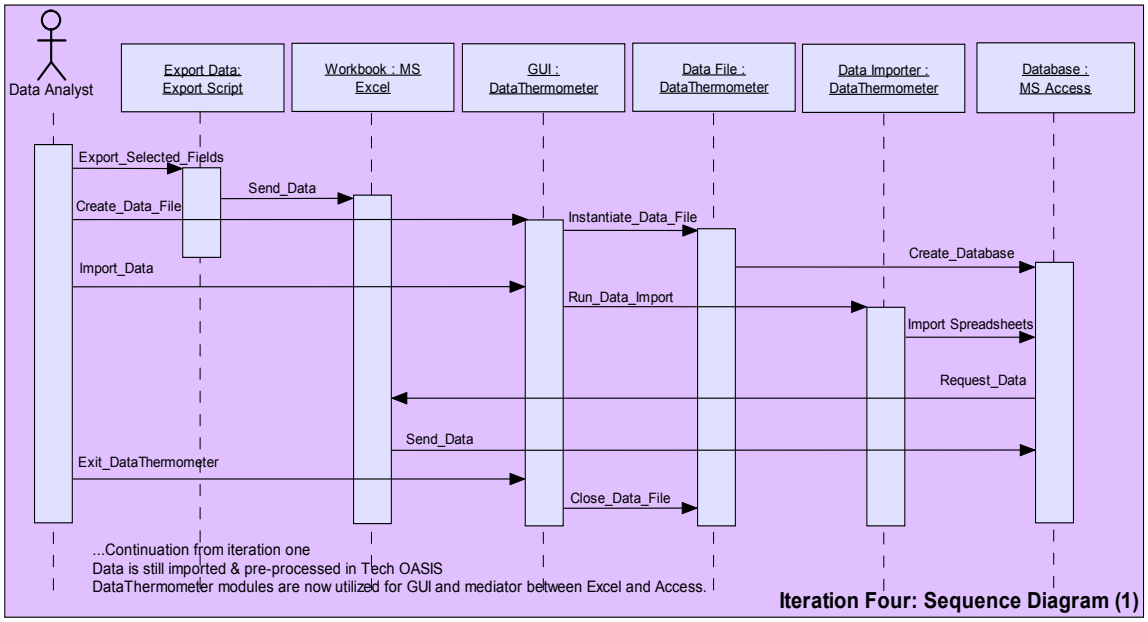


Figure 64. Iteration four sequence diagram one of two.

Figure 64 shows the following sequence of events:

1. The data is exported from Tech OASIS to a MS Excel workbook via export script.
2. The user selects “Create Data File” command in DataThermometer.
3. DataThermometer creates a new data file instance.
4. The data file object requests the creation of a new database.
5. The database is created.
6. The user selects the “Import Data from Excel” command.
7. The DataThermometer GUI instantiates the DataThermometer Data Importer object.
8. The Data Importer object executes MS Access functions to import MS Excel spreadsheets.
9. MS Access requests and receives the data from the MS Excel workbook.
10. The user selects the “Exit DataThermometer” command.
11. DataThermometer closes the opened data files and terminates.

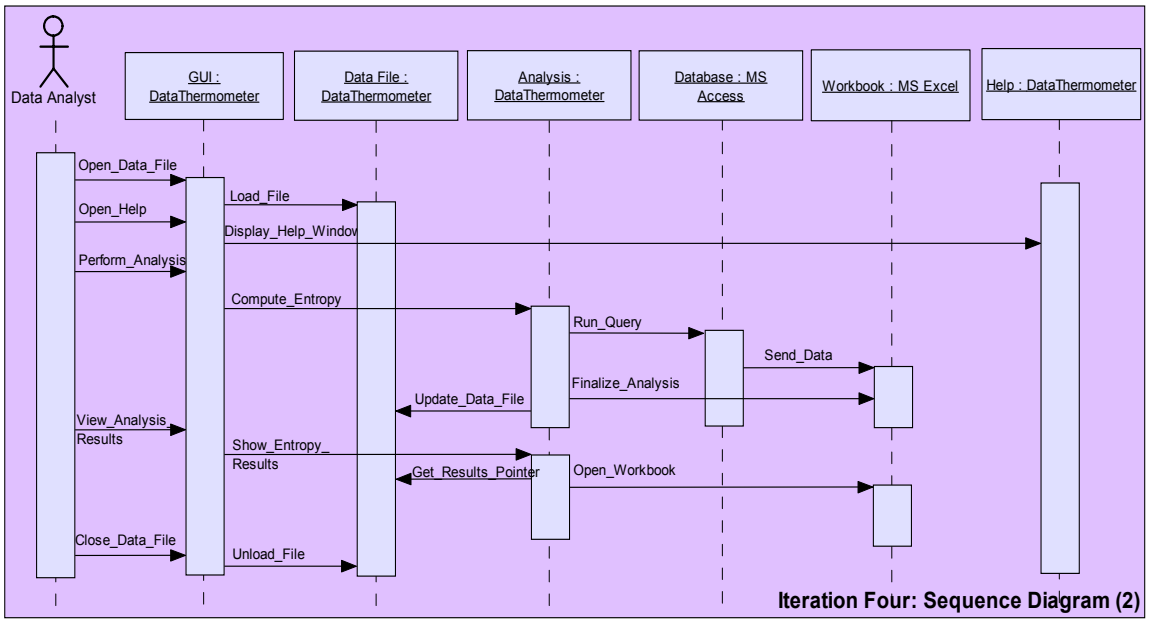


Figure 65. Iteration four sequence diagram two of two.

Figure 65 shows the following sequence of events:

1. The user selects the "Open Data File" command.
2. DataThermometer loads the datafile.
3. The user selects the "Open Help Documentation" command.
4. DataThermometer opens the help documentation in another window.
5. The user selects the "Perform Analysis Function" command and then chooses the "perform the entropy calculation" option.
6. DataThermometer instantiates the Analysis object.
7. The Analysis object performs operations on the MS Access database.
8. The MS Access database exports queries to MS Excel.
9. The Analysis object finalizes the calculations based on the exported queries.
10. The Analysis object updates the data file by noting the completion of the analysis function and setting the pointer to the results.
11. The user selects the "View Analysis Results" command and chooses to view the entropy calculation results.
12. DataThermometer instantiates the Analysis object.
13. The Analysis object sends a request to the data file for the location and name of the file that contains the results.
14. The analysis object opens the MS Excel workbook with the results.
15. The user selects the "Close Data File" command.
16. DataThermometer unloads the data file.

IV. RESULTS, CONCLUSIONS, AND FUTURE WORK

In this thesis three questions were investigated. The questions were presented in Chapter I and are the following:

1. Can the data analysis process be systemized?
2. What are the steps required to determine the temperature of a set of bibliographic records?
3. Can the learning curve of the process be reduced?

The answers to each of them lie in Chapter III and are explicitly stated in the following paragraphs.

Systemization of data analysis process proved to be successful. Based on the research and software development performed for this thesis, the first question was answered. By examining the four software development iterations presented in Chapter III, it can be seen how the process evolved. Figure 66 below shows the process flow as it existed during the final process development iteration. Notice how the chart parallels the KDD process chart described in Chapter II.

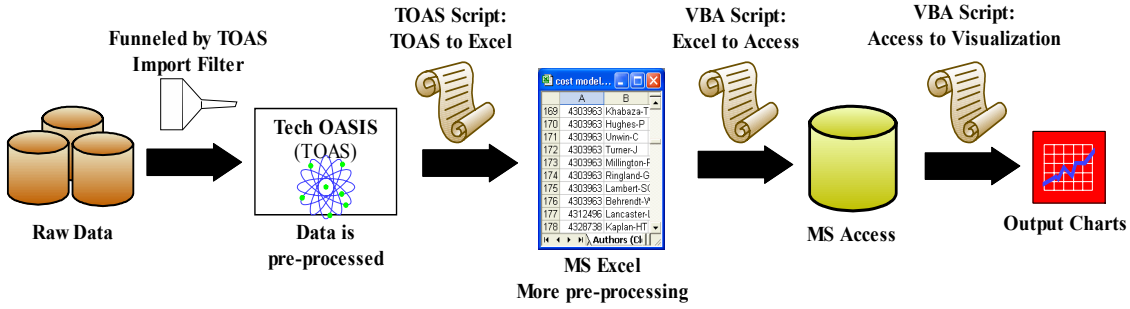


Figure 66. Iteration three overall process flow.

The steps required to determine the temperature of a set of bibliographic records are stated in Chapter III. There are two routes that can be taken when calculating the temperature. The calculation can be applied to a dataset's combined-descriptor term instances or just to a dataset's single-descriptor term instances. The exact steps involved to perform the temperature calculation on the combined terms are shown in Table 4,

where the steps are weighted based on ease of use because some steps require more effort than others. For example, setting up the queries requires substantially more effort than clicking a button to complete a task on a GUI. The scale ranges from 1 to 5 where 1 takes the least amount of effort and 5 takes the most.

Task	**Effort**
1. Obtain raw data from bibliographic databases.	3
2. Open Tech OASIS.	1
3. Create an import filter if one does not exist.	3
4. Import the raw data into Tech Oasis through the use of an import filter.	2
5. Remove the duplicate records in the dataset.	2
6. Standardize (clean) each of the data fields.	2
7. Send the data in the data fields to MS Excel along with the unique identifier for each record via Tech OASIS export script.	2
8. Apply the month and year information to each record based on the accession number by using the assign accession number macro.	2
9. Combine the terms together using the Term Combination macro.	2
10. Open MS Access.	1
11. Create a new database.	1
12. Import each term combination MS Excel worksheet into a separate MS Access database table.	5
13. Import the other MS Excel worksheets into MS Access (title, affiliation, authors, single descriptors).	3
14. Set up the MS Access queries as described in Iteration Three.	4
15. Export the results of each term combination query (co-occurrence matrix) to a MS Excel workbook.	2
16. Export the results of the other four queries – co-occurrence matrices of single-descriptors, authors, affiliations, and single-descriptor terms by affiliation to a separate MS Excel workbook.	3
17. Run the "Affiliation Distribution" macro on the workbook	2
18. Copy the "Affiliation Summary" sheet produced by the macro execution from the previous step to the workbook with the term combinations (see step 14).	2
19. Run the "Q Level Analysis" macro on the workbook with the term combination sheets and "Affiliation Summary" sheet.	2
Total Effort Required	***44***

Table 4. Steps and effort required to compute the temperature of the dataset with combined terms.

The last question also produces a favorable answer. The learning curve of the process can be reduced. Iteration four concerns packaging and integration, although not implemented, the iteration was well planned. The planning shows that adding one interface for all the importation and data analysis functions greatly reduce the number of tasks the user has to perform. To prove this, the steps listed below perform the same temperature calculation on the combined terms that is described above.

Task	**Estimated Effort**
1. Obtain raw data from bibliographic databases.	3
2. Open Tech OASIS.	1
3. Create an import filter if one does not exist.	3
4. Import the raw data into Tech Oasis through the use of an import filter.	2
5. Remove the duplicate records in the dataset.	2
6. Standardize (clean) each of the data fields.	2
7. Send the data in the data fields to MS Excel along with the unique identifier for each record via Tech OASIS export script.	2
8. Open DataThermometer.	1
9. Create a new DataThermometer datafile.	1
10. Use the "Import Data from Excel" command to import data into DataThermometer.	2
11. Select the "Perform Data Analysis" command.	1
12. Select "Temperature Calculation"	1
13. Select "View Data Analysis Results"	1
14. Select "Temperature Calculation"	1
Total Effort Required	***23***

Table 5. Steps and effort required to compute the temperature of the dataset with combined terms after integration.

As shown in the above tables, the total effort required goes from 44 to 23 with the use of DataThermometer's GUI to manage the handling of the data, calculation algorithms, and the analysis results.

An implementation methodology and software tool for an entropy based engineering model for an evolving system has been proven to work. The implementation methodology was applied to eight different sets of data and behaved as expected. Furthering this proof is the fact that the results obtained from the developed calculation

algorithms provided Dr. Michael Saboe an experimental reference to his doctoral dissertation. The resultant charts can be seen throughout his dissertation.

Future work involves the completion of the implementation of Iteration Four.

LIST OF REFERENCES

[1] *CCSU – Data Mining*. Central Connecticut State University. January 13, 2003 <http://www.ccsu.edu/datamining/>.

[2] Benning, S., Denning, M., Jaquint, C. Rusell, P., "Data Mining". University of Iowa. December 14, 2002 <http://www.biz.uiowa.edu/class/6k180_park/Student-Reports/sbenning/>.

[3] Rob, P., Coronel C., *Database Systems: Design, Implementation, and Management*, 5th ed., Course Technology, Inc., 2002.

[4] Benning, S., Denning, M., Jaquint, C. Rusell, P., "Data Mining". University of Iowa. December 14, 2002 <http://www.biz.uiowa.edu/class/6k180_park/Student-Reports/sbenning/>.

[5] Pilot Software Inc., "Discovering Hidden Value in Your Data Warehouse." 1998. March 3, 2003 <http://www.umich.edu/~cisdept/bba/320/1999/fall/pilot-software.html>.

[6] Han, J., Kamber M. *Data Mining, Concepts and Techniques*, Chapter 1. Morgan Kaufmann Publishers, 2000.

[7] Frawley W., and others, "Knowledge Discovery in Databases: An Overview," *AI Magazine*, pp. 213-228, Fall 1992.

[8] Gupta, V.R., "An Introduction to Data Warehousing." August 1997. January 28, 2003 <http://www.bac-atl.com/library/bi/dwintro.pdf>.

[9] Information Discovery Inc., "A Characterization of Data Mining Technologies and Processes." 1997, February 13, 2003 <http://www.datamining.com/dm-tech.htm>.

[10] Fayyad, U., Piatetsky-Shapiro, G. and P. Smyth, "The KDD Process for Extracting Useful Knowledge from Volumes of Data," *CACM*, 11th ed., v. 39, pp. 27-34, November 1996.

[11] Wisner, S., "Realities of Datamining." *Catalog Age* Jan 1999. January 30, 2003 <http://catalogagemag.com/ar/marketing_realities_datamining/>.

[12] Larman, C. *Applying UML and Patterns: An Introduction to Object-Orientated Analysis and Design and the Unified Process*, 2d ed., Prentice-Hall, Inc., 2002.

[13] Kulak, D. and Guiney, E., *Use Cases Requirements in Context*, Addison-Wesley., 2002.

[14] Wilson., Rauch T., Paige J. "Prototyping in the Software Development Cycle" 1992. December 7, 2002. <http://www.firelily.com/opinions/cycle.html>.

.

[15] Saboe, M.S., A Software Technology Transition Entropy Based Engineering Model, Ph.D. Dissertation (draft), Naval Postgraduate School, Monterey, California. March 2002.

APPENDIX A – REQUIREMENTS ELICITATION

The requirements of the system were obtained through a series of one on one communications with the primary researcher, Michael Saboe. The implementation followed an iterative process. Each iteration produced a prototype. Mike supplied the calculations, I implemented them so they could be applied to all of the datasets in his research, and finally I sent the results to him for review. Then, either revisions were made to the implementation to correct any errors or new functions were implemented to perform further data manipulation.

The implementation process continued for several months. From the process, several functions in the form of Microsoft Excel macros and Microsoft Access modules written in Visual Basic for Applications (VBA) evolved. It was then determined that these macros need to be placed inside a package with a GUI so they can be used to process future datasets and preserve the data processing methods.

The below interviews describe how I obtained the requirements for the macros:
Note: The interviews are paraphrased.

<u>Interview One: Establishing the problem, the beginning.</u>

I asked Michael Saboe what the primary purpose of the data analysis was. Mike told me the purpose was to provide an experimental reference to his dissertation on technology transfer. I asked several questions to gain some more insight into the analysis functions of the desired application. The following are some of the questions and answers:

Matt: How will this application provide an experimental reference to your dissertation?
Mike: The application will perform calculations on eight sets of data and generate results in the form of spreadsheets and charts.
Matt: Where does the data come from?
Mike: The data comes from online databases, specifically INSPEC.
Matt: You'll be supplying the data, right? What is in the data?
Mike: Yes, I'll give you the data. The data contains publication information on specific topics that I searched for. For example, I've searched for publication records using the following keywords: Ada, Java, Software Engineering, etc.
Matt: What publication information is in the data?
Mike: The key items are: title, descriptors, authors, affiliations, and accession number; a unique identifier INSPEC uses, which you could use to build relationships between the data fields.
Matt: What types of calculations are to be performed?
Mike: For starters, I need the entropy of each dataset's descriptors over time.
Matt: What do you mean, time?
Mike: Yearly intervals.
Matt: What's a descriptor?
Mike: A data field that gives the keywords used to index a paper.
Matt: Will you be supplying the equations for the calculations?
Mike: Yes, but implementation is up to you, as long as I have the results in a Microsoft Excel spreadsheet.

Matt: So you want me to parse the data and store the information somewhere?
Mike: Yes, I recommend you talk to Bob Watts in the NAC [National Automotive Center], he has a software tool called Tech OASIS that allows importation and manipulation of publication data from INSPEC.

The following are the key points from this interview:

- Raw data of publication records will be supplied.
- Tech OASIS can be used to import and clean the data.
- The equations to be implemented will be supplied to the developer.

Interview Two: The use of Tech OASIS

In this interview, I spoke with Bob Watts from the NAC concerning the use of Tech OASIS to import the data.

Matt: What does Tech OASIS allow me to do?
Bob: Tech OASIS allows you to import your data by using an import filter to parse the data fields. After that you can remove duplicate records and clean the data up.
Matt: How does an import filter work?
Bob: You setup the import filter to recognize data fields and have it extract the data in that field.
Matt: What do you mean, clean the data?
Bob: Tech OASIS can identify two items that are different but equal in a datafield.
Matt: Isn't that an oxymoron?
Bob: Not in this case, for example human-machine interface and human machine interface are listed as two different entries but are really the same thing.
Matt: So you recommend cleaning the dataset?
Bob: Yes.
Matt: How do I get the data to Microsoft Excel for further manipulation?
Bob: You can do this through Visual Basic automation scripts.

The following are the key points from this interview:

- Create an import filter to parse the data
- Remove duplicate data using Tech OASIS
- Clean the data using Tech OASIS
- Create an automation script to export the data to MS Excel.

Interview Three: Expanded Entropy Analysis

This interview took place after Tech OASIS was used to filter the raw data and perform the initial entropy analysis.

Mike: The initial analysis looks good. I have a couple of suggestions and requests.
Matt: What are they?
Mike: First, the cumulative entropy chart needs to show a power trendline.
Mike: And a summery sheet needs to be created.
Matt: What goes on the summary sheet?
Mike: Time intervals, cumulative entropy for each interval, and some calculations based on the trend line.
Mike: I also have a couple of new equations. First we'll need a new sheet to store and calculate a Lambda value and then the Lyapunov exponent needs to be calculated.

Matt: OK, send me some documentation describing the calculations and why they are important.

The following are the key points from this interview:

- Add a power trendline to the entropy chart.
- Generate a summary sheet
- Calculate lambda and the Lyapunov exponent

Interview Four: More Calculations

This next set of questions and answers describes additional calculations that need to be performed:

Matt: I've done the entropy calculation of descriptors over years, now what needs to be done?
Mike: Now that we've done this by years, we need to do it by month.
Matt: What else?
Mike: We need to know the number of authors, records, term instances, and affiliations per month.
Matt: You also mentioned something about combination of terms?
Mike: Yes, we need to know the combinations of doubles, triples, quadruples, ect. of all the terms in order to establish the vocabulary of the dataset. After you obtain the vocabulary you need to compute the entropy of the vocabulary, the temperature and the pressure.

Matt: You'll give me the equations to do this right?
Mike: Yes, first determine the vocabulary and we'll move further from there.

The following are the key points from this interview:

- Perform calculations by month.
- Obtain the number of authors, records, terms instances, and affiliations per month.
- Establish the vocabulary of the dataset by getting the combination of terms
- Compute the entropy, temperature, and pressure of the dataset.

Interview Five: Packaging the data manipulation functions inside a GUI,

The Creation of DataThermometer:

The set of questions and answers below describe the problem of preserving the functions created.

Matt: After processing all the data you need for your dissertation, what is the next step.
Mike: We need to use the methods you've implemented to process future datasets.
Matt: Well, at the current time, I'm the only one that knows how to do this.
Mike: This is true, that's why you should put a front-end on these functions.
Matt: Are you suggesting a GUI?
Mike: Yes, a graphical user interface would work, provided there were detailed instructions included with it.
Matt: Right, using Visual Basic would speed this along since most of the code is compatible with this language.

The following are the key points from this interview:

- The methods of processing the data must be able to be used on other datasets.
- Create a GUI using Visual Basic to preserve the data manipulation functions.
- Create documentation outlining the processes used to complete the data analysis.

From this last interview, the following problem statement became clear:

A solution needs to exist that will preserve the calculation and data processing methods derived from Michael Saboe's doctoral dissertation. This solution must have a user interface which will let the user perform, at minimum, the following tasks:

Select the data processing methods
Generate and store results

From this problem the project proposal became apparent.

Project Proposal

Project Title: DataThermometer: an application to measure the stability and complexity of a technology and the transfer channel.

Background: The application is based on the technology transition model developed by Michael Saboe for his doctoral dissertation.

Main customer: Researchers

Goal: To create an application, with a user interface, that will take in a dataset and output intensive and extensive quantities.

Problem statement:

Elements	Description
The problem of	duplicating the research results of this doctoral dissertation
affects	researchers
the results of which	is wasted effort implementing methods that have been already developed.
Benefits of	a new application to address the problem include: • Ability to extend the research into other areas. • Saves research time, and • Research provides a useful tool instead of just a paper.

APPENDIX B – SAMPLE CALCULATIONS

Entropy definition

Michael Saboe describes entropy and how it relates to information theory in his doctoral dissertation (also presenting in Chapter I):

> Entropy as a concept can readily be seen as logical entropy (think of it as a measure of uncertainty, noise, non-signal, process inefficiencies, the percentage of work resulting in defects and requiring rework, etc) and physical or thermodynamic entropy (i.e. mixed-up-ed-ness, disorder, disorganization, etc), which is the quantity of energy not available to do work. Logical entropy is Shannon's entropy (S_H) as defined by Shannon on his treatise on communication theory (Shannon 1948). Shannon's theory says that the entropy of an information source measures how well its behavior (e.g. the next symbol in a sequence it produced) can be predicted.
>
> The definition of entropy here is related to the definition of entropy in thermodynamics. What follows is a basic review of entropy in information theory after Shannon (1948). Let X be a discrete random variable with alphabet Ξ and a probability mass function $p(x)=\Pr\{X=x\}$, $x \in \Xi$. $p(x)$ and $p(y)$ refer to two different random variables and are in fact two different probability mass functions $p_x(x)$ and $p_y(y)$.
>
> The definition of information entropy is:
>
> $$S_H(X) = -\sum_{x \in \Xi} p(x) \log_2 p(x)$$
>
> S_H is the entropy measured in bits, and the log is base 2. Log_2 will be assumed throughout unless otherwise noted. For example, the entropy of a fair coin toss is 1 bit. The convention of $0^+ \log 0^+ \to 0$ is used, which comes from continuity since $x \log x \to 0^+$, as $x \to 0^+$. The base of the log is two for the natural units of information entropy as developed by Shannon (Shannon 1948). The entropy is a function of the distribution of X. It does not depend on the actual values taken by the random variable X, but only on the probabilities. (Saboe, 2001)

Entropy Calculation Equations and Example:

The formula used in this calcuation is the following:

$$S_H(X) = -\sum_{x \in \Xi} p(x) \log_2 p(x) \qquad (1.1)$$

Where p(x) is the probability of term usage. The probability of term usage is the cumulative number of a single term's instances up to the given time interval divided by the number of terms instances of all the terms up to the given time interval. The following two tables give an example of the calculation:

Table B-1

Sample Data

Term	1989	1990	1991
A – # of instances	2	3	5
B – # of instances	1	5	18
Local sum	3	8	23
Cumulative sum	3	11	34

Table B-2

Sample Entropy Calculation Example

Term	1989	1990	1991
Entropy of A	$-(2/3) * \log_2(2/3) = 0.3900$	$-(5/11) * \log_2(5/11) = 0.5170$	$-(10/34) * \log_2(10/34) = 0.5193$
Entropy of B	$-(1/3) * \log_2(1/3) = 0.5283$	$-(6/11) * \log_2(6/11) = 0.4770$	$-(24/34) * \log_2(24/34) = 0.3547$
Cumulative entropy	0.9183 (a + b)	0.9940	0.8740

Predicted Entropy Calculation

The predicted entropy value for a time interval is calculated using the trend-line power equation (the least squares fit through points):

$$y = cx^b \tag{1.2}$$

where c and b are constants. The time interval replaces x. An example from the Ada dataset follows in Table B3.

Percent Error of Actual vs. Predicted

Percent error of actual vs. predicted is calculated using the formula below and an example follows in Table B3.

$$\text{Error} = \frac{\text{Predicted} - \text{Actual}}{\text{Actual}} \tag{1.3}$$

Table B-3

Predicted Entropy Calculation and Error Example

Time T	Slice	Actual: Cum Entropy	Error (Act vs. Pred) y = 4.7404x^.1489	Predicted: 5 years of data
1	1979	4.48385619	5.71%	4.48385619
2	1980	5.406900167	-2.86%	5.406900167
3	1981	5.805013635	-3.93%	5.805013635
4	1982	6.057909749	-3.94%	6.057909749
5	1983	6.082181413	-1.11%	6.082181413
6	1984	6.106601538	1.19%	6.179377493
7	1985	6.355700897	-0.53%	6.321976128
8	1986	6.52682382	-1.21%	6.44815784
9	1987	6.549095131	0.19%	6.561546835
10	1988	6.611519798	0.80%	6.664665264

Note: First 5 intervals under the predicted column are copied from actual.

Time Interval Derivative Calculation

Computing entropy from a process of transferring information, from using the Lyapunov exponent and lambda, is also described in Michael Saboe's doctoral dissertation:

> We compute entropy two ways. One is from data, which is acquired experimentally. Using the cumulative approach seems appropriate since the messages are persistent and available to all of the future researchers to examine. The other is from a model of the process of transferring information. The experimental entropy data is related to the information we know about a topic. We refer to this as Shannon's entropy (S_H). The data S_H is gathered over time steps k. We perform regression on this data and have as a result a function that is of the power law form (e.g., $y=mx^b$, where m is the slope and b is the intercept). We also have a model of a non-linear dynamical system. The Lyaponuv exponent of a map gives the sensitive dependence upon initial conditions that is characteristic of chaotic behavior.
>
> The eigenvalue of the Jacobian of the finite difference equations representing the dynamical system is also related to entropy. The Jacobian matrix is simply the derivative of a p-dimensional map function F. Saboe uses the form $x_{n+1} = F(x_n)$, where x is a p-dimensional vector. In fact, the relationship is through the Lyaponuv number.
>
> The Lyaponuv number and this relationship is defined as

$J_n = [J(x_n)\ J(x_{n-1})..\ J(x_1)]$

where $J(x)$ is the Jacobian matrix of the map

$J(x) = (\partial F/\partial x)$ with $j_1(n) \geq j_2(n) \ldots \geq j_p(n)$ are the magnitudes of the eigenvalues of J_n. The Lyaponuv numbers are

$\lambda_i = \lim_{n\to\infty} [j_1(n)]^{1/n}$, $i = 1,2,\ldots,p.$

The Lyaponuv exponent is the smallest, positive, real *n*th root taken. The Lyaponuv exponent is the log of the Lyaponuv number.

The Du(T) and Du_(T-c) calculations for the "Entropy Lambda" sheet are based on the derivative of the trend-line's equation from the cumulative entropy graph. The derivative of the trend-line's equation is taken and then the time interval replaces x. The following is the equation used:

$$\frac{dy}{dx}\left[y = cx^b \right] = cbx^{(b-1)} \tag{1.4}$$

A usage example from the Ada dataset follows in Table B4.

Table B-4

Time Interval Derivative Calculation Example

Time T	**Du_(T)**	**du_(T-c)**	**du_(T-2)**	**du_(T-5)**
1	0.70152			
2	0.388653426	0.70152		
3	0.275126706	0.388653426	0.70152	
4	0.215320284	0.275126706	0.388653426	
5	0.178040011	0.215320284	0.275126706	
6	0.152424645	0.178040011	0.215320284	0.70152
7	0.133664638	0.152424645	0.178040011	0.388653426
8	0.11929092	0.133664638	0.152424645	0.275126706
9	0.107900992	0.11929092	0.133664638	0.215320284
10	0.098637046	0.107900992	0.11929092	0.178040011
11	0.090943882	0.098637046	0.107900992	0.152424645
12	0.084445719	0.090943882	0.098637046	0.133664638
13	0.078878805	0.084445719	0.090943882	0.11929092
14	0.074052371	0.078878805	0.084445719	0.107900992
15	0.069824897	0.074052371	0.078878805	0.098637046

Note. $y = 4.7404x^{0.1489}$

du(T) = (4.740*0.148)*T^(0.148-1)

du(T-c) = (4.740*0.148)*(T-c)^(0.148-1)

Lambda Calculation

The lambda calculation is dependent on the time interval derivative calculations.

The equation to calculate lambda is:

$$\lambda_0 = (\beta f'(x))^{1/3} \tag{1.5}$$

Where $f'(x)$ is substituted with:

$$f'(x_0) = \frac{\partial y}{\partial x} = \frac{\dfrac{du_{(t-c)}}{dt}}{\beta \dfrac{du_{(t-c)}}{dt} + \dfrac{du_{(t)}}{dt}} \tag{1.6}$$

substituting for $f'(x)$ we get .

$$\lambda_0 = \left(\beta \frac{\dfrac{du_{(t-c)}}{dt}}{\beta \dfrac{du_{(t-c)}}{dt} + \dfrac{du_{(t)}}{dt}} \right)^{1/3} \tag{1.7}$$

The values from the time interval derivative equation (1.4) are placed into (1.7) with varying β values (e.g. 0.1, 0.2, 0.5, 0.75). Table B5 shows an example of the lambda calculation.

Table B-5

Lambda Calculation Example

Time T	Cum Entropy	Du_(T)	C_y_10%	Lambda_ β 10%_y	β 10%_y
1	4.48385619	0.70152			

2	5.406900167	0.388653426	4.87216694	0.534733226	0.1
3	5.805013635	0.275126706	5.306648158	0.498365477	0.1
4	6.057909749	0.215320284	5.574025253	0.483884497	0.1
5	6.082181413	0.178040011	5.60612135	0.476060063	0.1
6	6.106601538	0.152424645	5.635448868	0.47115267	0.1
7	6.355700897	0.133664638	5.887915573	0.467785324	0.1
8	6.52682382	0.11929092	6.061493127	0.465330693	0.1
9	6.549095131	0.107900992	6.085633441	0.46346169	0.1
10	6.611519798	0.098637046	6.14952886	0.461990938	0.1
11	6.64290191	0.090943882	6.182098576	0.460803334	0.1
12	6.725985485	0.084445719	6.266161229	0.459824255	0.1

Note. "C_y_10%" is found from "Cum Entropy" minus "Lambda_ β 10%_y"

Lyapunov Exponent Calculation

The Lyapunov exponent calculation depends on the trend-line equation from the map of entropy at time steps k and k+1. The derivative is taken the same way as in equation (1.4). Once the derivative is found the time interval is replaced for x. A usage example from the Ada dataset follows in Table B6.

Table B-6
Lyapunov Exponent Calculation Example

Time T	Cumulative Entropy K	Cum_K+1	Lyapunov Exp J'(k,k+1) = 0.724*1.720 k^(0.724-1)
1	4.48385619	5.406900167	0.823021444
2	5.406900167	5.805013635	0.781579695
3	5.805013635	6.057909749	0.766403215
4	6.057909749	6.082181413	0.757435961
5	6.082181413	6.106601538	0.756600505
6	6.106601538	6.355700897	0.755764222
7	6.355700897	6.52682382	0.74747023
8	6.52682382	6.549095131	0.742009203
9	6.549095131	6.611519798	0.741311905
10	6.611519798	6.64290191	0.739373453
11	6.64290191	6.725985485	0.738407755
12	6.725985485	6.817516503	0.735878946

Note. $y = 1.7208x^{0.7241}$
$dx = 0.724*1.720)K^{(0.724-1)}$

APPENDIX C – SABOE DISSERTATION EXCERPTS

The following pages were taken from Dr. Michael Saboe's doctoral dissertation and are referenced in this thesis's body.

Each of these subsets represents a possible way that a researcher may find this message. Often, as we know, we use only elements of some research, that is single or double or more sets of terms. Each of these are legitimate accessible states of the message. The higher q-level terms can be viewed as higher level concepts. You can see by inspection that it is not possible using this approach to have a q=3 term without filling the lower level q-level states. At some point the combination of a q=2 or q=3 set of terms can take on meaning as a primitive terms in and of itself. These higher q-level sets take on the meaning of a higher level of abstraction. They can then be considered representations of a concept, which may be replaced by new single terms.

At that point, it becomes a q=1 set. We would expect that the higher level q sets will exhaust when they become more and more frequently used. This seems to be consistent with the abstraction discussion (Whitehead 1910), and learning models (Newell 1980) (See 11. Abstraction, p91, and 10. Learning Curves, p90 and Chapter III). Shannon (1948) illustrated this using a telegraphy notation where, a birth or death was simply represented by a few terms. The receiver understood that those few terms implied that a baby boy was born on a certain date, and other appropriate details. We do the same thing when we learn. We follow an economy of symbols and the principle of least effort (Zipf 1949) discussed earlier.

We could look at the entire message of the publication (the article or report), and we could, in fact, look at every term in the publication and determine the frequency of occurrence of the set of sets of terms. If we were looking at every term in a message or report, we could also populate the lower half of the matrix. This would permit the determination that {AB} was different from {BA}, because {AB} has A preceding B and the reverse is true in the case of {BA}. This is how the analysis would be done for a free text study.

For the purposes of this experiment, we are using a bibliographic record, and only examining the descriptors. In a descriptor field, we would not expect the term to be entered more than once, and the order is not significant in the data source used in this study. We do assume that the terms in the descriptor field are representative of the topics covered in a message. Further, that the message terms in this field are symbolic of the

future (open system) similar to the rates for the historic (closed) subsystems. This permits the design of a desired solution in the form of an engine.

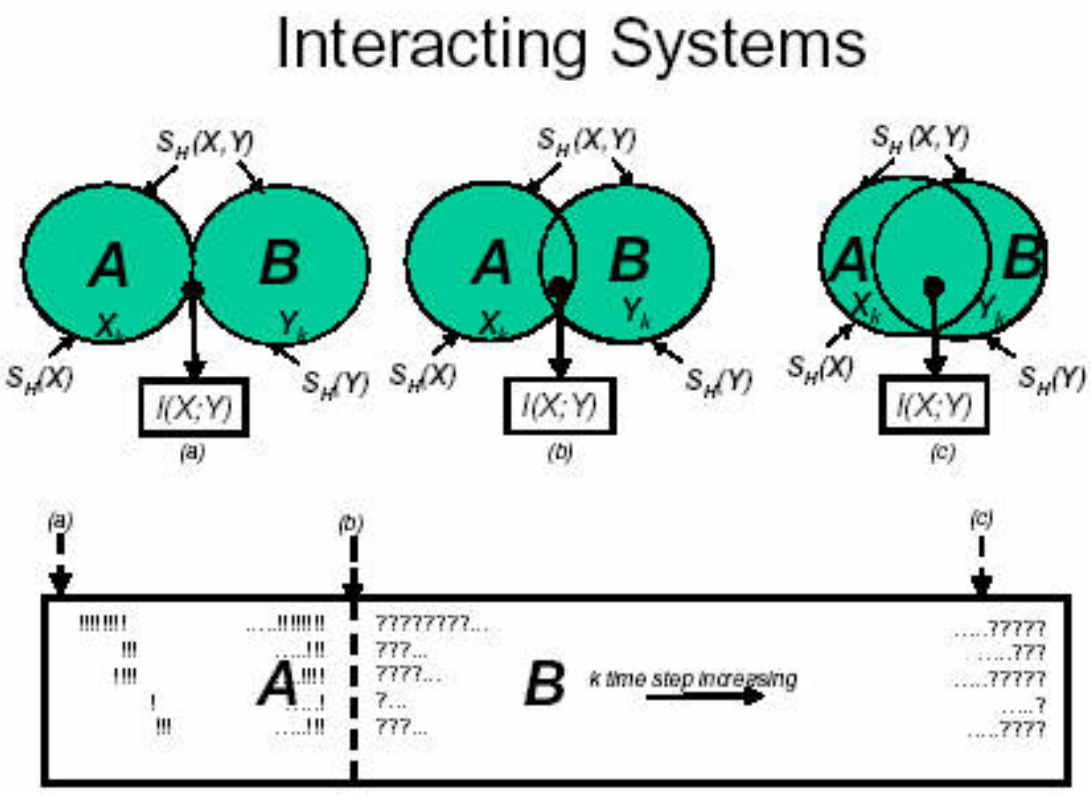


Figure III-10 Interacting Systems A and B

In Figure III-12[14], we see that as system A expands, the number of terms discovered increases, at the same rate that the number of terms undiscovered decreases. This model satisfies our conservation principle for extensive quantities.

Next, in Figure III-13, we examine the entropy relationship. The horizontal line at the top of the figure is the joint entropy of the system. Since this is a closed system, this is not changing, however, the internal distribution will change. That entropy related to subsystem A will increase as the are more and more choices to make in order to get complete information. Subsystem B will decrease from a high entropy (all of the unknown terms) to a lower entropy as there becomes less and less left to be discovered.

[14] The charts in this section represent initial data to illustrate the general relationships. Actual

The lower curve shows the mutual information. When the distance between the center of the two probability masses, or subsystems, decreases, there is a higher correlation.

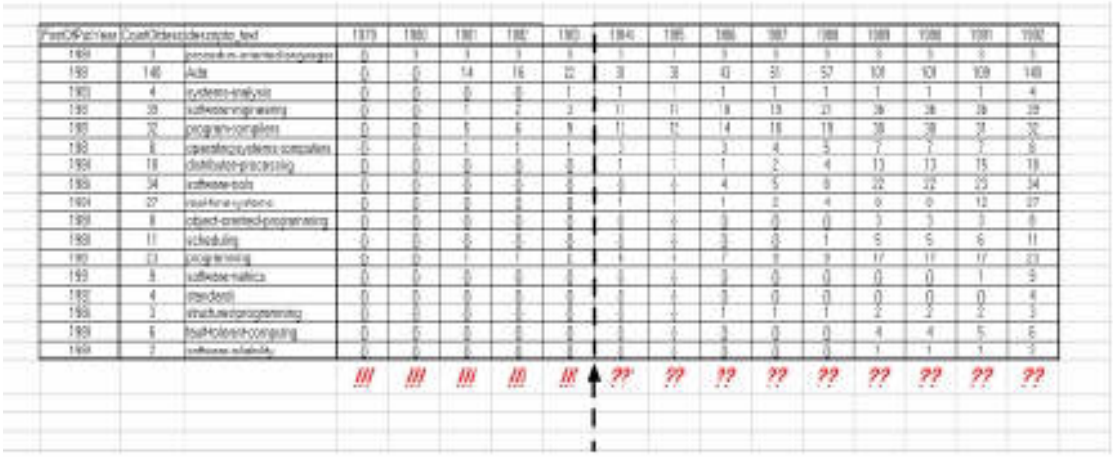


Figure III-11 Subset of an alphabet in two interacting systems !!! and ???

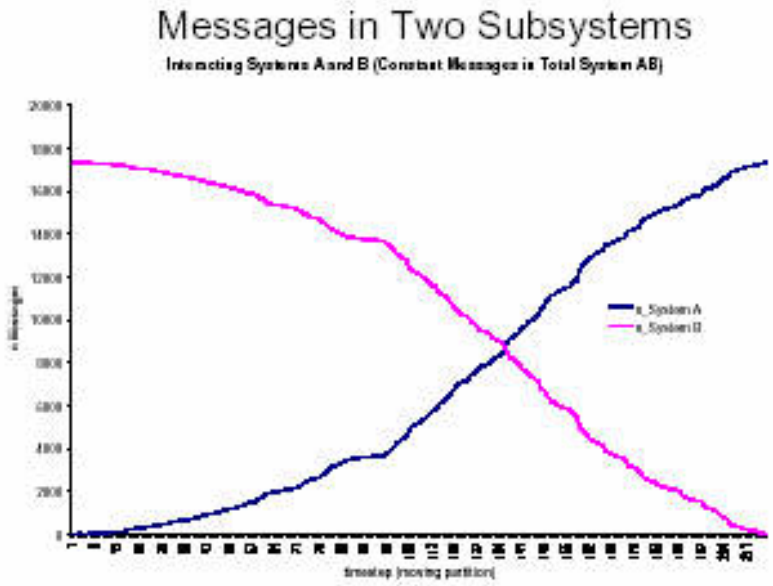


Figure III-12 Messages in two subsystems

equations for a specific technology are shown in Chapter IV, and the appendix.

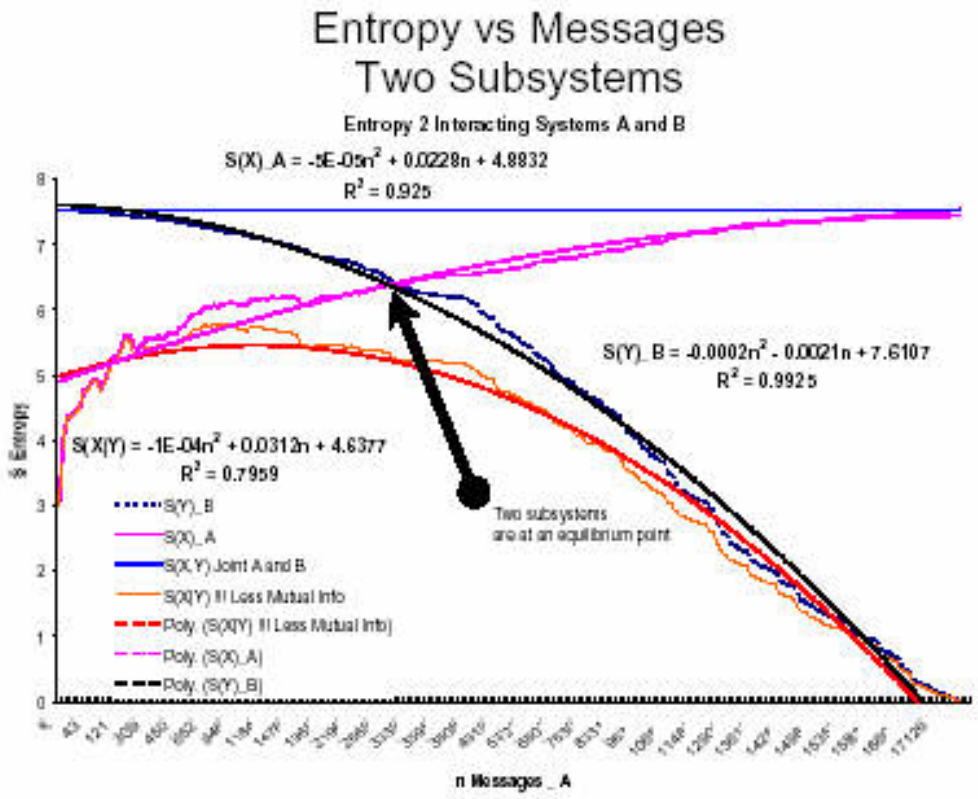


Figure III-13 Entropy vs. Messages Two interacting Systems

Following reasoning similar to that used in statistical, and condensed particle physics (Schroeder 2000) (Fraundorff 2000), we can find some useful relationships. The slope of the curves of the two subsystems gives us some important information about thermal equilibrium. Recall from the canonical ensemble discussion of free energy, that the temperature T is the parameter controlling free energy, or the conserved property. In this case of messages, we can write

$$\frac{1}{T} = \frac{\Delta S_H}{\Delta n} \qquad (3.25)$$

So the temperature is related to slope of the change in entropy verses change in messages curves. When the curves in the figure cross over, the system is at an

equilibrium point. Let's look at a general relationship that shows the increase in one system is related to the negative slope or, the decrease in the other.

$$\frac{\Delta S_A}{\Delta n_A} = -\frac{\Delta S_B}{\Delta n_A} \qquad (3.26)$$

The incremental change in S_A divided by the change in n_A messages, is equal to the change in entropy, S_B for system B again compared to the change in the conserved quantity, in this case n_A. Rewriting we get

$$\frac{\Delta S_A}{\Delta n_A} + \frac{\Delta S_B}{\Delta n_A} = 0 \qquad (3.27)$$

The second term has a B in the numerator and A in the denominator. Δn_A is the same as $-\Delta n_B$, since what we discover in messages is the same as what is removed from the undiscovered system. We can rewrite this for a system at equilibrium as

$$\frac{\Delta S_A}{\Delta n_A} = \frac{\Delta S_B}{\Delta n_B} \qquad (3.28)$$

The thing that is the same for both systems when they are at thermal equilibrium is the slope of the entropy message graph. This slope must somehow be related to the temperature of the system. The 2nd law of thermodynamics tells us that the conserved property will tend to flow *into* the subsystem with the *steeper* entropy vs. message graph, and *out* of the object with the *shallower* entropy vs. message graph (Schroeder 2000 p87).

According to Schroeder, the former "wants to" gain the free conserved property (messages) in order to increase its entropy. If there is an imbalance between the two subsystems, the latter doesn't so much "mind" losing a few messages (since the entropy will not decrease much. A steep slope must correspond to a *low* temperature, while a shallow slope corresponds to a *high* temperature.

Now we can see in the lower curve of Figure III-14, the relationship of the temperature (the right hand y-axis) of sub-system A as the partition moves over the time steps. More activity increases the temperature. The temperature is measured in degrees as we would in a physical system; however, these degrees are developed from

information units. This is "the" fundamental temperature unit developed from the relationship of entropy, and the conserved quantity.

Note that there are temperature fluctuations. This is consistent with Prigogine's observation about evolving systems. A dynamical system will help explain these fluctuations.

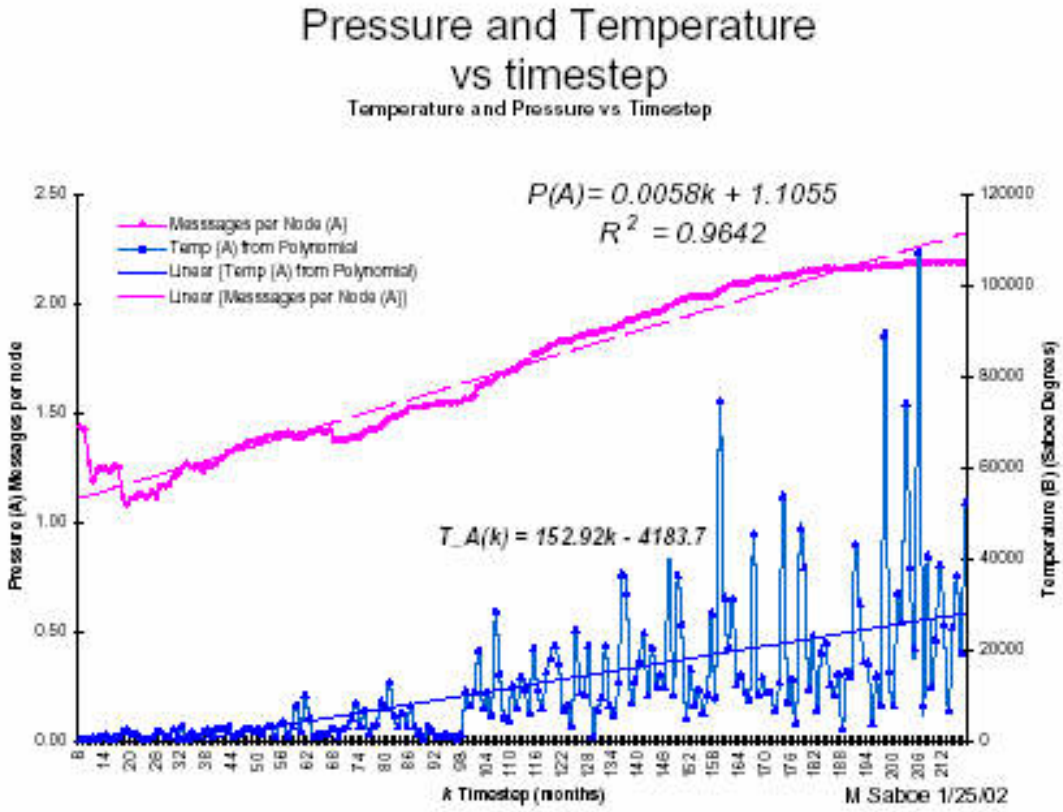


Figure III-14 Pressure and Temperature °Saboc© vs. time – two interacting systems

Pressure is defined as the <messages> processed per node, where the <messages> represent the average in the time step per node. The important observation is not necessarily the form of the equations or the goodness of fit, rather, that ***the pressure can be seen to increase as the temperature increases***. While messages are not physical molecules as in a thermodynamic system, they seem to behave as a gas might, as the temperature goes up the pressure goes up.

Figure III-15 shows the relationship directly between pressure and temperature. This was developed by taking the curves from Figure III-14 and setting them both equals to *k*. Then the Pressure *P(T)* as a function of temperature is determined.

$$P(k) = m_P + b_P \tag{3.29}$$

$$\frac{P(k) - b_P}{m_P} = k \tag{3.30}$$

Similarly, solve for *k* as a function of *T*.

$$T(k) = m_T k + b_T \tag{3.31}$$

$$\frac{T(k) - b_T}{m_T} = k \tag{3.32}$$

Then we get

$$\frac{P(k) - b_P}{m_P} = \frac{T(k) - b_T}{m_T} = k \tag{3.33}$$

$$P(k) = \frac{m_P}{m_T}(T(k) - b_T) + b_P \tag{3.34}$$

When plotted in Figure III-15 is the tight set of points indicating as temperature increases, pressure increases. Figure III-15 shows the raw data points as well. These fluctuate around the *P(T)* calculated data, would be expected.

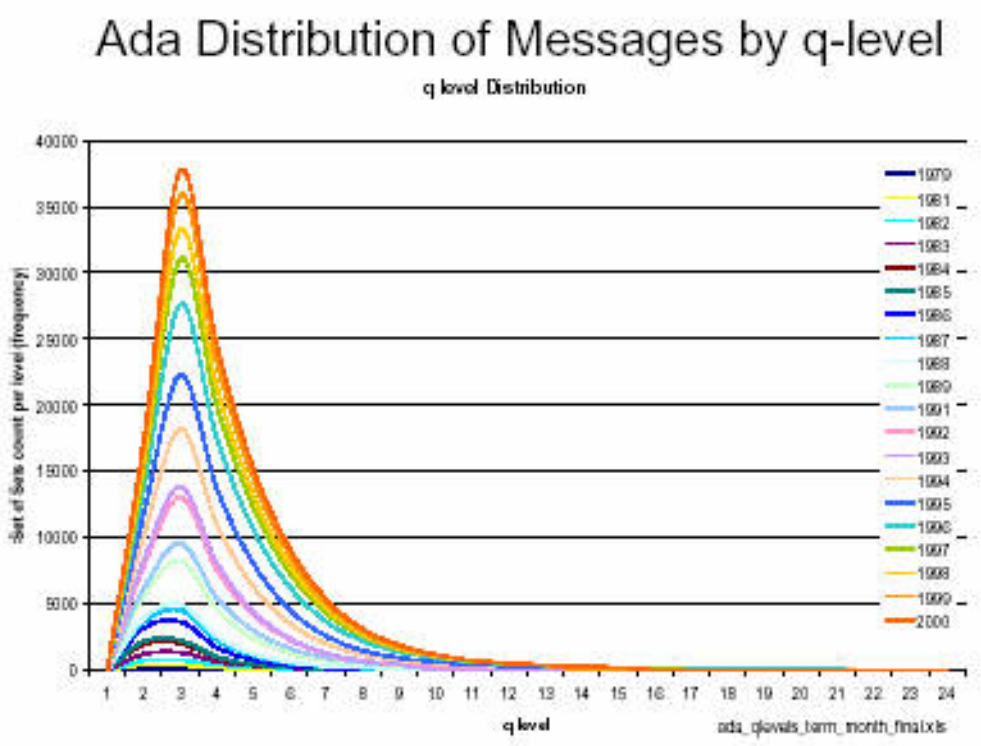


Figure III-18 Set of sets distribution over time steps by q-level

8. Technology Transfer Channel Elements

We consider two cases. The deterministic case represents the microscopic level of the model in the system, and the stochastic case represents the macroscopic system view. So far, we have only addressed the macroscopic case. The deterministic case would occur at the micro level in a program, or a system made up of nodes consisting of a family of machines. A stochastic system consists of a population, coarsely partitioned at the macroscopic level. This is a system made up of a social environment consisting of people and organizations. The *TechTx* models address the more general case of the stochastic system of nodes consisting of people, organizations and machines.

We define the community, the macro structure, as a set of performers that produce output. An organizational is made up of a set of the performers with which they are affiliated. We can think of the micro level in terms of the performers. The organizational level is in between the macro and micro levels and can be thought of as an ensemble of affiliated performers. We can observe individual output from the data. Each record contains primitive messages published by a performer, x_{i_k} contributes information to the community. This is defined as follows.

$$\overline{X} = \bigcup_{i=1}^{p} \overline{X}_i \text{ is the } community \tag{3.36}$$

$$\overline{X}_i = \left\{ x_{i_1}, x_{i_2} \ldots x_{i_k} \right\} \text{ where } x_{i_1}, x_{i_2} \ldots x_{i_k} \text{ are the } performers \text{ of the } i^{th} \text{ } organization \tag{3.37}$$

$$\overline{X}_i \text{ is the } i^{th} \text{ organization, and } i = 1..p \tag{3.38}$$

The output entropy is allocated from the message to individual author subset performers from the empirical data. This micro level is then summed up and allocated to the to the affiliated organizational level. The organizations are banded based on a distribution of the cumulative number of published messages.

We consider a family of nodes (machines, and people – the atomic level), making up organizations (the molecular level), and a community (macro level). In a band, we assume all of the nodes have the equivalent properties, i.e. each organizational node, comprised of performing author nodes, are statistically equivalent. Figure III-19 and Figure III-20 illustrates a node taking information in as input *S(X)*, performing some transformation, $F(X_k)$, to produce more messages (work products). Part of the output is expanding the mutual information *I(X;Y)* intersection of the Venn diagram, and part is augmenting the vocabulary. This augmentation is the conditional probability *S(Y|X)*, as we saw from equation (3.15).

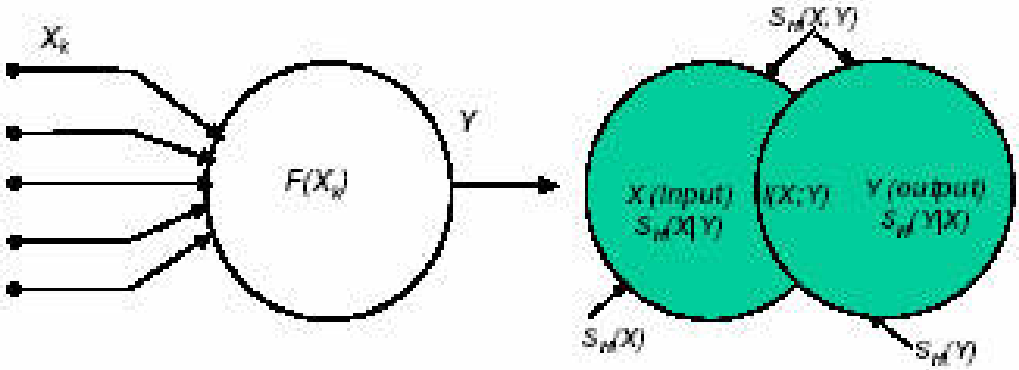


Figure III-19 Input being converted via a transfer function to output

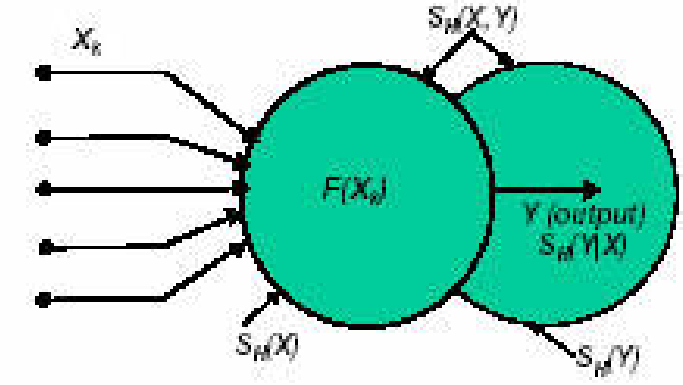


Figure III-20 Node transform of Input to Output.

The initial band determination is computed based on the accumulation of experience of executing tasks, i.e. publishing messages. The most prolific performers are banded together based on the average number of messages produced over the period examined. Later the learning, or performance index is computed for each band at every time step from the beginning of the data set to the (current) performance time step. An example of the distribution is shown in Figure III-21.

We will perform a coarse partitioning of the performing organizations into four bands. Further, partitions are possible, however this is sufficient to demonstrate the approach. The "A" band consists of all of the organizations that were beyond 3σ in the rate of production of messages in the sample for a given technology. The "B" band are the organizations in the 3σ partition. The "C" band contains the organizations with a message production history in the 2σ partition, and the "D" band are all of the organization below 2σ in performance.

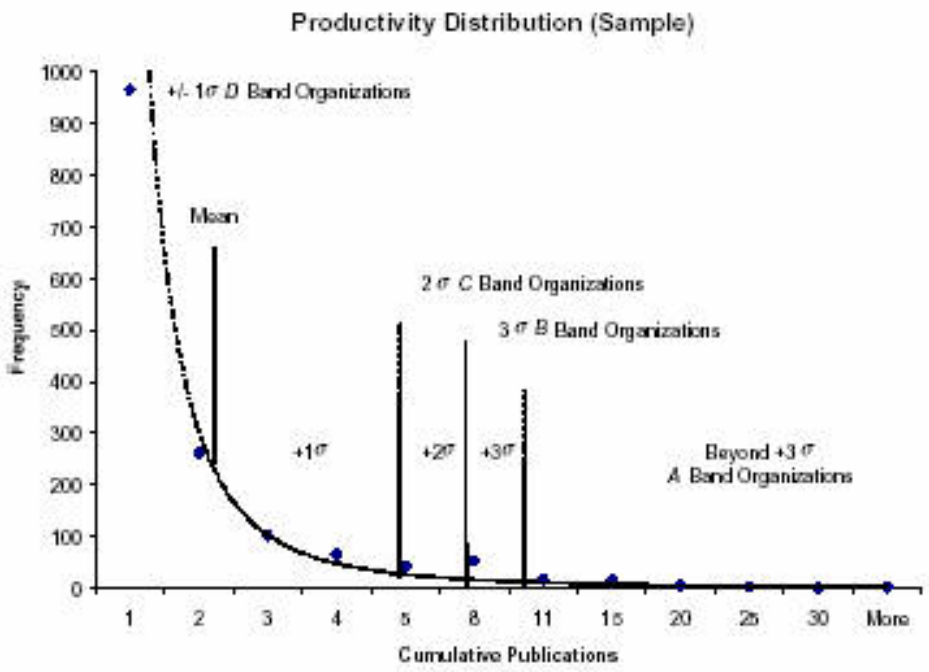


Figure III-21 Organization Distribution into Cumulative Task Performed Bands

Our problem is to realize, or at least to approximate, a given system, which we call the true system, by a model. We adjust the parameters values based on a number of examples provided by observation of the true system.

The analyses of the partitions can proceed exactly as the analysis of the macro level community. This is the beauty of the partitioning. We only have to be cautious of combining bands when the counts of terms, (multiplicity of states) are "local" to the band under examination. We count messages in a band and develop the probabilities, and hence the entropy of the band is based on the total number of messages in the band. In order to aggregate bands, we consider this entropy the band's contribution to the total (all bands) entropy. There is an entropy contribution simply resulting from the partition. This contribution varies every time step based on the internal organization of the messages, constituent terms, and nodes.

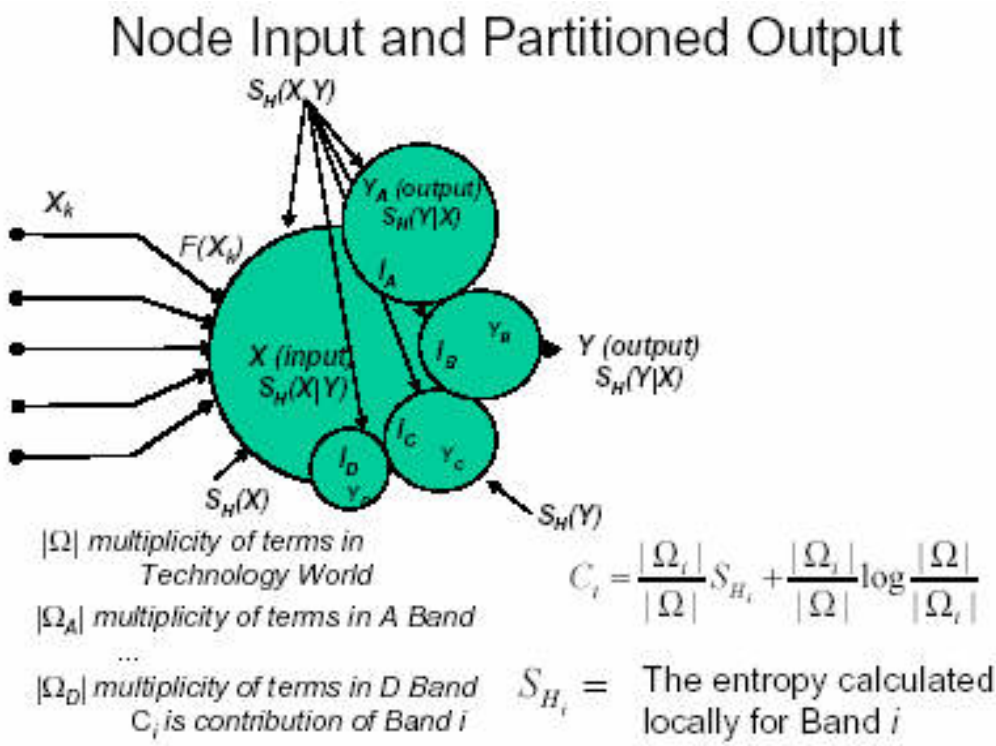


Figure III-22 Partitions of output into bands. Contribution to the Community

Each band, i, provides a contribution, C_i to the community entropy. The local band entropy S_{H_i}, must be scaled based on the multiplicity Ω_i of terms in the band to the multiplicity Ω of terms in the world. The community, which is sometimes referred to as the technology's "world" entropy is the sum of the contributions.

$$S_{H_{world}} = \sum_{i=1}^{n\ bands} C_i \qquad (3.39)$$

$$\text{where } C_i = \frac{|\Omega_i|}{|\Omega|} S_{H_i} + \frac{|\Omega_i|}{|\Omega|} \log \frac{|\Omega|}{|\Omega_i|} \qquad (3.40)$$

This relationship permits aggregation of previous results on a subset of a community with more information later without having to rerun the entire world and all previously analyzed bands. All that is required is the count of the instances of terms in a

band and the count of the number of instances of terms in the world, augmented by these terms.

Later extensions to be considered would address all of the various combinations of author nodes producing a message. For example, the x_{i_1} performers could be represented as combinations of authors producing a record (which as was pointed out, is broken down into its primitive messages at various q-levels). Additionally, we could assume that if there are three authors on a record, they represent 2^3 possible author subsets – nodes. Each subset is a legitimate combination producing the messages. This distribution develops in exactly the same way as the term distribution of sets of sets as developed. The ability to calculate the contribution with a ratio of the local system instances to the microstates of a larger or smaller system, it was often useful to count instances of states. By computing the entropy locally, these chunks can be combined with other subsystems often with out additional computation.

opportunity to relate the two entropy measures, S_H and S_B since they are related to the same information system of messages N. We are dealing with the same information flows, hence the same system, so this seems reasonable. Recall S_B is related to Lyapunov's exponent λ, which comes from the eigenvalue j.

We found the relationship of messages verse time step in Figure III-8 was very satisfactorily modeled as a linear equation for this technology set. (It could be different for other technologies, this is why we have dealt with the relationships in terms of functions, eigenvalues and derivatives.) In this case, the derivative of the linear model reduced to a constant in equation (3.72) as noted earlier.

Now instead of using an average, or guess for β, it is computed directly. To compute β, the amount of information that a node consumes which persists in time, both equations are a function of timestamp, so we can solve $S_B(k)=j_k$, $j_k = b_j k^{m_j}$ and $S_H(k)$ $S_H = b_S k^{m_s}$, for k. Setting them both equal to each other, we can solve for β (S_B,S_H).

$$\left(\frac{S_k}{b_S}\right)^{\frac{1}{m_S}} = k \qquad (3.74)$$

and

$$\left(\frac{j_k}{b_j}\right)^{\frac{1}{m_j}} = k \tag{3.75}$$

which yields

$$j_k(S_k) = b_j\left(\frac{S_k}{b_S}\right)^{\frac{m_j}{m_S}} \tag{3.76}$$

$$S_k(j_k) = b_S\left(\frac{j_k}{b_j}\right)^{\frac{m_S}{m_j}} \tag{3.77}$$

Here the subscripts s refers to slope and intercept terms of the Shannon entropy equation, and the subscript j is referring to similar terms in the S_B, bakers transform equation.

Before we do, let's explore the relationship to temperature from the discrete, micro, model. Earlier, using a macroscopic approach, we showed that temperature increases, or decreases with increasing or decreasing pressure on a node respectively. In a physical system, we can address temperature in of entropy and conserved property, let's see that this is true for this discrete, micro formulation as well.

In Figure III-30, we see on the left-hand side, that there is a transfer function that converts X input, or some percentage of the available persistent input, into Y, output. This is really made up of two parts as seen on the right. We can use a Venn diagram as introduced earlier. Extensive properties like messages are additive. Probabilities are multiplicative. This also applies to the entropy.

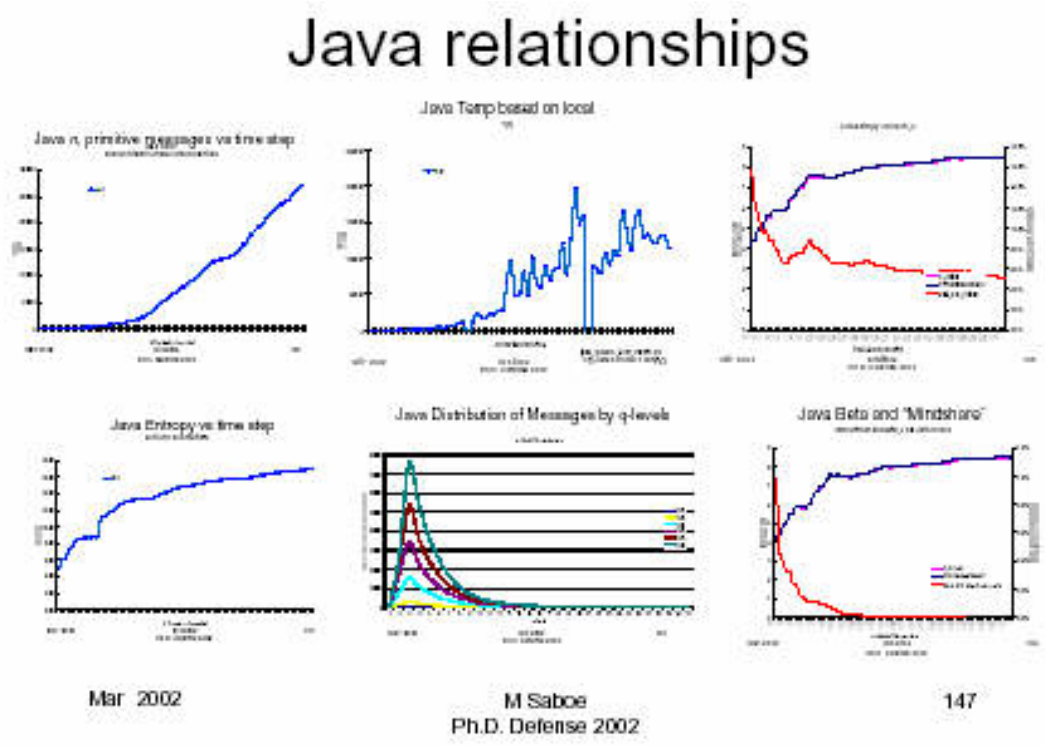


Figure IV-15 Java Relationships

For an early Ada example seen in Figure IV-16, we can observe that both curves, the curve for the Lyapunov exponent and Shannon's entropy have the same power law form. By observation, we see both of the entropy measures as a function of time step. S_H,, the information theory entropy measure is on the left *y-axis*, and the S_B which comes from the eigenvalue of the micro control model (hence in the range of 0 to 1), is on the right hand *y*-axis. The scales were adjusted to easily see that both curves are of the same form. In addition, we can see for this early data set, that the R^2 values are reasonable, at 0.968 for system level entropy and 0.96 for the bakers transformation *f*. We can see that as the system entropy stabilizes, the eigenvalue of the feedback control dynamical system is also stabilizing.

Initially, to determine the *form* of the functions, the average value $\beta \approx 10\%$ was used. This was done by iterative guesses of a fixed β. This approximation of β was used

to satisfy the macroscopic rate of change of entropy. This suggests that we have the right form of the dynamical system control model matched to the macroscopic system model. This also suggests that the model does approximate the observed conditions.

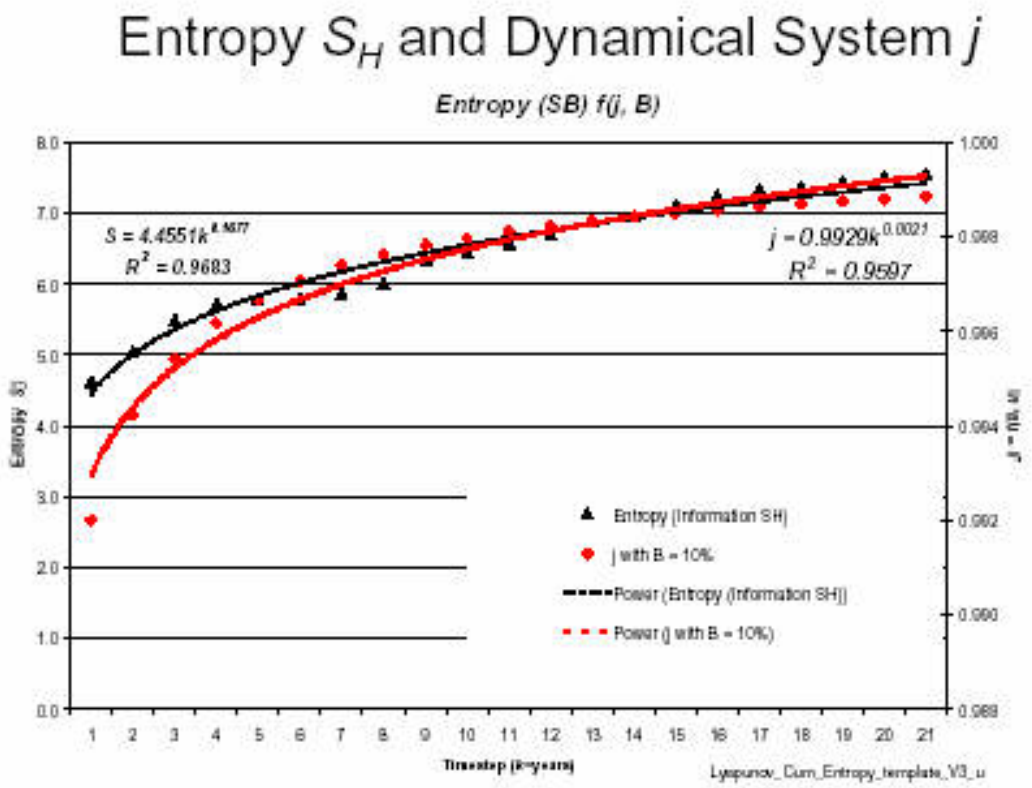


Figure IV-16 Macro Equilibrium S_H and Eigenvalue j Stabilization

From (3.77), which develops Shannon entropy now in terms of j_k, which we know from (3.70) is a function of β. At this point β was adjusted until the entropy (eigenvalue) of the discrete model matched the macroscopic entropy of the information theoretic model. In each time step, the tolerance on the two methods of computing the entropy were matched to within 0.1%. This is seen in Figure IV-17. The upper curves (the two are superimposed) represent entropy converging at the same timestamps for the system. The lower curve represents β, which changes over time. The secondary y-axis, on the right gives β as a percentage.

Software Requirements Specification

for

DataThermometer

Version 0.5 approved

Prepared by Matthew J. Behnke

United States Army

Tank Automotive and Armaments Command

June 2003

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Matthew J. Behnke	6/15/02	Created SRS document	0.5

Introduction

Purpose

The purpose of this document is to specify the requirements of the initial version of the DataThermometer application.

Document Conventions

This document will give most of the requirements at a high-level of abstraction. The detailed requirements will be appended in the future.

Intended Audience and Reading Suggestions

This document is intended to be read by the software developers and the stakeholders discussed in DataThermometer's Vision and Scope document section 4.1.

Product Scope

Please refer to the Vision and Scope document.

References

Title	Author	Version	Date
DataThermometer Requirements Elicitation Document	Matthew J. Behnke	N/A	4/2/02
DataThermometer Vision and Scope Document	Matthew J. Behnke	.5	4/2/02
DataThermometer Use Case Document	Matthew J. Behnke	.5	4/2/02

Overall Description

Product Perspective

DataThermometer is a new self-contained, stand-alone, application.

Product Functions

Feature #	Feature
1.	Imports data from Microsoft Excel spreadsheets.
2.	Stores the data in a relational database.
3.	All functionality is performed by the user through DataThermometer's graphical user interface (GUI).
4.	User-selected data processing methods are applied to the data.
5.	The results of the data analysis are in the form of Microsoft Excel spreadsheets.
6.	DataThermometer keeps track of the data processing methods already ran on a dataset.

User Classes and Characteristics

See stakeholders in section 4.1 of the Vision and Scope document.

Operating Environment

DataThermometer must run in a Microsoft Windows 95+ environment and must be able to interact with Microsoft Office applications, Excel and Access version 97+.

DataThermometer's minimum hardware requirements are a Pentium-III or equivalent with 256 megabytes of RAM and 100 megabytes of hard drive space.

Design and Implementation Constraints

The following are design and implementation constraints:

Minimum Hardware Requirements:
Intel Pentium III or equivalent, 256 megabytes of RAM, 100 megabytes of hard drive space.

Language:
Microsoft Visual Basic

Data Storage:
DataThermometer must store parsed data in a Microsoft Access relational database. Results of the DataThermometer's analysis must be in the form of Microsoft Excel spreadsheets and charts.

Maintenance:
The finished product will be maintained by the development team.

User Documentation

User manuals and on-line help will be supplied as part of the initial release.

Assumptions and Dependencies

See Vision and Scope document section 2.3.

External Interface Requirements

User Interfaces

DataThermometer is intended to be used by technically skilled users.
The user's goals are to import data, perform data analysis, and review the results.

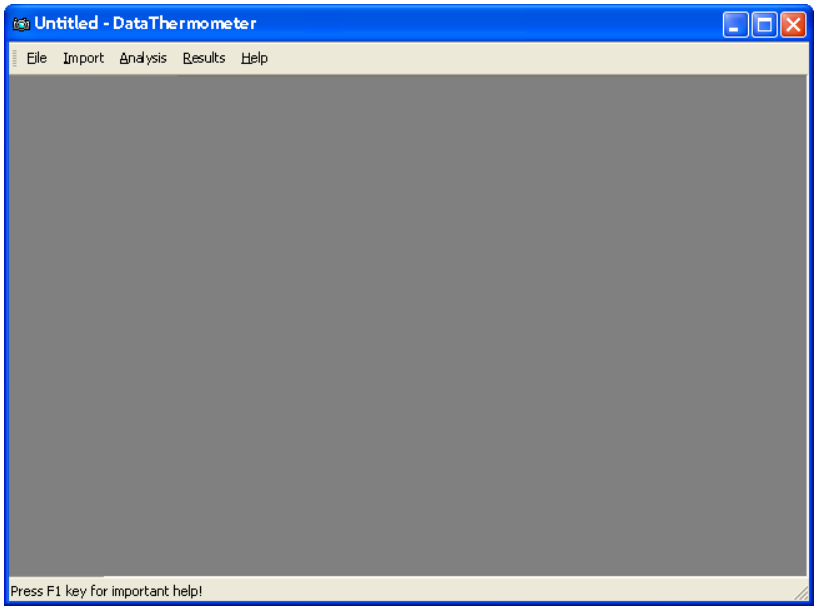
The following items are always shown in the top tool bar of the window:

- File - contains subcommands such as Open data file, Close data file, New Data File, and Exit program.
- Import – contains subcommands to build an import filter and import data using the filter and a command to import directly from Excel.
- Analysis – contains subcommands to perform data analysis methods on the data
- Results – After a data analysis method has been performed on the dataset subcommands become available in this menu to view the results of each analysis.
- Help – Selecting this command will open the help screens.

Keyboard shortcuts will be implemented for each menu

The window area is reserved for user input / selection of any details a sub command requires.

The following figure represents a sample of what the user interface will look like:



Hardware Interfaces

N/A

Software Interfaces

The interfaces between software components will be handled with Visual Basic functions to tie DataThermometer to a Microsoft Access 97 database. Visual Basic functions will also store data analysis results in Microsoft Excel 97+ spreadsheets.

Communications Interfaces

N/A

System Features

The functional requirements are shown below. To see the stimulus / response sequences see the use cases in the DataThermometer's Use Case Document.

DataThermometer Functional Requirements

Table 1: User Interface

1.1	All functionality must be able to be performed by the user through DataThermometer's graphical user interface (GUI).
1.2	The GUI shall contain high-level commands in the tool bar with subcommands underneath.
1.2.1	The high-level commands shall be contained under the following menu headers: File, Import, Analysis, Results, and Help.
1.3	DataThermometer must allow the user to cancel an operation during a state that requires user input. When an operation is canceled the program must return to an idle state.

Table 2: High-Level Functions

2.1	Users must be able to perform operations on a data file.
2.2	Users must be allowed to import data from Microsoft Excel spreadsheets.
2.3	Users must be allowed to perform data analysis on the dataset.
2.4	Users must be able to view the results of the previous performed data analysis.
2.5	Users must be allowed to access DataThermometer's help documents from within the application.

2.6	Users must be allowed to exit DataThermometer.
2.6.1	DataThermometer must close any open data files before exiting.

Table 3: Data File

3.1	DataThermometer must store information about a dataset in a DataThermometer Data File.
3.2	Users must be allowed to open an existing data file.
3.3	Users must be allowed to close an open data file.
3.4	Users must be allowed to create a new data file.
3.4.1	DataThermometer must prompt the user to name the datafile.
3.4.2	A newly created DataThermometer Data File must have the extension ".dt"
3.4.3	DataThermometer must store the name of the data set. The user is prompted for the name of the dataset.
3.4.4	DataThermometer must create the Access database used for the dataset when a new datafile is created.
3.4.5	DataThermometer must set the name of the database to: <the name of DataThermometer's datafile>-db.mdb
3.4.6	DataThermometer must store the name of the database in the data file.
3.5	DataThermometer must store an identifier telling which analysis functions have been ran on a dataset in the datafile.
3.5.1	DataThermometer must store the names of the Microsoft Excel files that contain the results of the analysis functions that have been ran in the datafile.
3.5.2	DataThermometer must store the date that an analysis function ran on the dataset in the datafile.

Table 4: Import Data

4.1	DataThermometer must be able to import data from Microsoft Excel spreadsheets.
4.2	DataThermometer must import data from Excel into a Microsoft Access database
4.3	DataThermometer must let the user select the sheets in an Excel spreadsheet to import into the Access database. The default is all sheets.
4.4	DataThermometer must allow the user to change the name of the imported Microsoft Access table. The default value is the name of the sheet in the Excel spreadsheet.

Table 5: Analysis

5.1	The user must be able to select from a list of data processing methods that can be applied to the dataset.
5.2	The results of the data analysis must be stored in the form of Microsoft Excel spreadsheets and charts.
5.3	Analysis must be based on the calculation methods previously implemented.

Table 6: Results

6.1	DataThermometer must allow the users to view the results of a data analysis function if that function has been ran on the dataset.

Other Nonfunctional Requirements

Performance Requirements

N/A, TBD

Safety Requirements

N/A, TBD

Security Requirements

N/A, TBD

Software Quality Attributes

Correctness – The methods implemented need to be correct in order to provide a useful product.

Maintainability – The software needs to be written in a way that will support maintainability and evolution of the product.

Reliability – DataThermometer, like any application, needs to be reliable. Meaning data analysis methods will be able to process to their full extent without error.

Usability – The application needs to apply to a wide domain of data. If this can be accomplished then one of the main goals will be satisfied.

Appendix A (of SRS): Glossary

Dataset – a collection of data.

DataThermomemter Data File – File used to store information about a dataset (e.g., name of dataset, whether or not data has been imported, type of analysis methods ran on the dataset, etc.).

Import Filter – A set of rules that govern how DataThermometer parses a data file when importing data.

THIS PAGE INTENTIONALLY LEFT BLANK

Initial Use Cases

for

DataThermometer

Version 0.5 approved

Prepared by Matthew J. Behnke

United States Army
Tank Automotive and Armaments Command

June 2002

Revision History

Name	Date	Reason For Changes	Version
Matt Behnke	6/21/02	Created Initial Use-Cases	0.5

Table of Contents

Use Case ID:	1		
Use Case Name:	Open Data File		
Created By:	Matt Behnke	Last Updated By:	Matt Behnke
Date Created:	6/16/2002	Date Last Updated:	6/16/2002

Actor:	User, DataThermometer, Data file
Description:	DataThermometer prompts the user to select a data file.
Preconditions:	DataThermometer must be loaded and in an idle state.
Postconditions:	Data file loaded.
Normal Course of Events:	1. The user selects the Open Data File command. 2. DataThermometer checks to see if a data file is already loaded. 3. A file is loaded. DataThermometer asks the user if it should close the file. 4. User responds "Yes" to close the file. 5. DataThermometer closes the file. 6. DataThermometer displays directory tree. 7. User navigates directory tree. 8. User selects a file. 9. User confirms file to be loaded. 10. DataThermometer successfully reads the file
Alternative Courses:	3. A data file isn't loaded. 4. DataThermometer displays directory tree. 5. User navigates directory tree. 6. User selects a file. 7. User confirms file to be loaded. 8. DataThermometer successfully reads the file 6. User doesn't find desired file. 7. User cancels Open Data File 8. DataThermometer can not read file. Gives an error.
Exceptions:	TBD
Includes:	Close Data File (Use Case ID: 2)
Assumptions:	
Notes and Issues:	Exceptions will be determined at a later date.
Requirements:	1.1, 1.3, 2.1, 3.1, 3.2

Use Case ID:	2		
Use Case Name:	Close Data File		
Created By:	Matt Behnke	Last Updated By:	Matt Behnke
Date Created:	6/16/2002	Date Last Updated:	6/16/2002

Actor:	User, DataThermometer, Data file
Description:	DataThermometer closes the data file.
Preconditions:	DataThermometer must be started, be in an idle state, and a data file must be loaded.
Postconditions:	Data file is closed.
Normal Course of Events:	1. User selects Close Data File 2. DataThermometer closes the data file.
Alternative Courses:	
Exceptions:	TBD
Includes:	
Assumptions:	
Notes and Issues:	Exceptions will be determined at a later date.
Requirements:	1.1, 2.1, 3.1, 3.3

Use Case ID:	3		
Use Case Name:	New Data File		
Created By:	Matt Behnke	Last Updated By:	Matt Behnke
Date Created:	6/16/2002	Date Last Updated:	6/16/2002

Actor:	User, DataThermometer, Data file
Description:	DataThermometer creates a new data file.
Preconditions:	DataThermometer must be loaded and in an idle state.
Postconditions:	Data file loaded.
Normal Course of Events:	1. The user selects the Open Data File command. 2. DataThermometer checks to see if a data file is already loaded. 3. A file is loaded. DataThermometer asks the user if it should close the file. 4. User responds “Yes” to close the file. 5. DataThermometer closes the file and prompts for the new data file’s name. 6. DataThermometer displays directory tree. 7. User navigates directory tree. 8. User selects a folder and enters a file name for the data

	file. 9. DataThermometer creates the data file. 10. DataThermometer tells the user to enter the name of the dataset. 11. User gives a name for the dataset 12. DataThermometer stores the data set's name in the data file. 13. DataThermometer creates the Access database for the dataset, named: the name of the datafile + "-db.mdb"
Alternative Courses:	3. A data file isn't loaded. 4. DataThermometer prompts for the new data file's name. 5. User navigates directory tree. 6. User selects a folder and enters a file name for the data file. 7. DataThermometer creates the data file. 8. DataThermometer tells the user to enter the name of the dataset. 9. User gives a name for the dataset 10. DataThermometer stores the data set's name in the data file. 11. DataThermometer creates the Access database for the dataset, named: the name of the datafile + "-db.mdb" 9. & 7. DataThermometer can't create data file. Gives an error. --- 11. User doesn't enter a name for the dataset. (Just hits enter) 12. DataThermometer sets dataset name to equal the data file's filename 13. DataThermometer creates the Access database for the dataset, named: the name of the datafile + "-db.mdb"
Exceptions:	TBD
Includes:	N/A
Assumptions:	
Notes and Issues:	Exceptions will be determined at a later date.
Requirements:	1.1, 1.3, 2.1, 3.1, 3.4, 3.4.1-3.4.6

Use Case ID:	4		
Use Case Name:	Exit DataThermometer		
Created By:	Matt Behnke	Last Updated By:	Matt Behnke
Date Created:	6/16/2002	Date Last Updated:	6/16/2002

Actor:	User, DataThermometer, Data file
Description:	DataThermometer shuts down.
Preconditions:	DataThermometer is running and idle.
Postconditions:	DataThermometer stops.
Normal Course of Events:	1. User selects Exit DataThermometer command. 2. DataThermometer prompts, "Are you sure you want to exit?" 3. User confirms, "Yes". 4. DateThermometer checks if a data file is open. 5. A data file is open 6. DataThermometer closes the file. 7. DataThermometer shuts down.
Alternative Courses:	5. No data file open. 6. DataThermometer shuts down. --- 3. User doesn't confirm, "No" 4. DataThermometer returns to idle.
Exceptions:	TBD
Includes:	Close File (Use Case ID: 2)
Special Requirements:	
Assumptions:	
Notes and Issues:	Exceptions will be determined at a later date.
Requirements:	1.1, 1.3, 2.6, 2.6.1

Use Case ID:	5		
Use Case Name:	Import Data from Excel		
Created By:	Matt Behnke	Last Updated By:	Matt Behnke
Date Created:	6/16/2002	Date Last Updated:	6/16/2002

Actor:	DataThermometer, User, Microsoft Excel File
Description:	Imports data from Excel into an empty data file. Stores the data in an Microsoft Access database. An example of the import screen can be found below in Figure 1.
Preconditions:	DataThermometer has been started, a new data file has been loaded.

Postconditions:	Data is added to the new data file.
Normal Course of Events:	1. User selects Import Data from Excel. 2. DataThermometer makes sure a new file has been created. 3. A new data file has been created. 4. DataThermometer prompts for the Excel file name. 5. User selects file. 6. DataThermometer asks which sheets in the Excel file to choose and allows changes to be made to the name of the imported table. (See Figure 1 below) 7. User places a check next to the sheet names to be imported and makes any changes to the table names. 8. User selects the import command. 9. DataThermometer imports the selected worksheets from the Excel file to the Access database.
Alternative Courses:	3. A new data file hasn't been created. 4. DataThermometer tells the user to create a new data file in order to use this function. 5. User cancels the selection of the Excel file. 6. DataThermometer stops the import command and returns to an idle state. 8. User selects the cancel import command. 9. DataThermometer stops the import command and returns to an idle state.
Exceptions:	TBD
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	Exceptions will be determined at a later date.
Requirements:	1.1, 1.3, 2.2, 4.1, 4.2, 4.3, 4.4

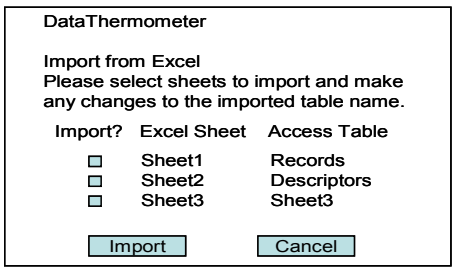


Figure 1. Example import screen.

Use Case ID:	6		
Use Case Name:	Perform Data Analysis		
Created By:	Matt Behnke	Last Updated By:	Matt Behnke
Date Created:	6/16/2002	Date Last Updated:	6/16/2002

Actor:	DataThermometer, User, Data file
Description:	The user selects a data processing method to be ran on the data file's data stored in it's database. During this process the data is manipulated and fed through calculations. The results of the calculations are stored in an Excel spreadsheet. DataThermometer stores which data analysis methods have been run on the data file and the date in which it ran.
Preconditions:	DataThermometer must be running and be in an idle state. A DataThermometer data file must be loaded and contain data.
Postconditions:	Data analysis results are created in a Microsoft Excel spreadsheet. A link to the spreadsheet is stored in DataThermometer's data file.
Normal Course of Events:	1. User selects Perform Data Analysis 2. DataThermometer prompts the user to select the data analysis function that will be performed. (functions map to analysis functions from previous interations). 3. DataThermometer obtains the data it needs from the Access database. 4. DataThermometer processes the data. 5. DataThermometer creates an Excel file and stores the results in it. 6. DataThermometer creates a link to the location of the resulting Excel file. 7. DataThermometer stores the fact that the selected data analysis method has been ran on the dataset, along with the date it ran.
Alternative Courses:	
Exceptions:	TBD
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	Exceptions will be determined at a later date.
Requirements:	1.1, 1.3, 2.3, 3.5, 3.5.1, 3.5.2, 5.1, 5.2

Use Case ID:	7		
Use Case Name:	View Data Analysis Results		
Created By:	Matt Behnke	Last Updated By:	Matt Behnke
Date Created:	6/16/2002	Date Last Updated:	6/16/2002

Actor:	DataThermometer, User, Excel file containing the results of the selected function.
Description:	After a data analysis function has been performed a link to the spreadsheet file with the results is stored. When users want to see the results of a particular analysis they select this command. This command opens the Excel file.
Preconditions:	DataThermometer must be running and be in an idle state and the data analysis function must have been ran.
Postconditions:	If it exists, the Excel file with the analysis results is opened.
Normal Course of Events:	1. User selects View Results of Data Analysis. 2. DataThermometer checks to see if any data analysis has been performed. 3. Analysis has been performed. 4. DataThermometer ….. 5. DataThermometer attempts to open the Excel file with the results based upon the stored file name. 6. The Excel file is opened.
Alternative Courses:	3. The function hasn't been performed. 4. DataThermometer tells the user to run Data Analysis Function xxx before trying to view the results. 5. The Excel file can't be found.
Exceptions:	TBD
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	Exceptions will be determined at a later date.
Requirements:	1.1, 1.3, 2.4, 6.1

Use Case ID:	8		
Use Case Name:	Open Help Documentation		
Created By:	Matt Behnke	Last Updated By:	Matt Behnke
Date Created:	6/16/2002	Date Last Updated:	6/16/2002

Actor:	DataThermometer, User
Description:	DataThermometer's help screen is opened.
Preconditions:	DataThermometer must be running and in an idle state.
Postconditions:	Help screen is opened.
Normal Course of Events:	1. User selects Open Help Documentation. 2. DataThermometer displays the help documentation.
Alternative Courses:	
Exceptions:	TBD
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	Exceptions will be determined at a later date.
Requirements:	1.1, 2.5

APPENDIX F – INSPEC DATA FIELDS

INSPEC records are divided into the following fields, listed in alphabetic order. Highlighted fields are **limit** fields.

AA	Author Affiliation
AB	Abstract
AI	Astronomical Object Indexing
AN	**Accession Number**
AU	Author
AV	Availability
CC	**Classification Codes**
CD	Conference Details
CI	Chemical Indexing
CL	Copyright Clearance Center Code
CO	CODEN
CP	**Country of Publication**
CS	Copyright Statement (*)
DE	Descriptors
DN	Document Number (*)
DOI	Digital Object Identifier (*)
DS	Dissertation Submission Date
DU	Document Collection URL (*)
ED	Editor
IB	ISBN
ID	Identifiers
IS	ISSN
LA	**Language**
MD	Description of Unconventional Medium
MN	Material Identity Number
NI	Numerical Data Indexing
OP	Original Patent Details
PA	Patent Assignee
PD	Patent Details
PF	Patent File Date
PI	Patent Priority Date
PR	Price
PY	**Publication Year**
RF	**Number of References**
RN	Report Numbers
RT	**Record Type**
SC	SICI (*)
SF	**Subfile**
SK	Sort Key
SO	Source
ST	SICI of Translation (*)
SU	Subject Terms (DE and ID)
TI	Title
TL	Translator
TR	**Treatment Codes**
UD	**Update Code**
UR	Universal Resource Locator (*)

(*) This field is for display only; you cannot search in this field.

THIS PAGE INTENTIONALLY LEFT BLANK

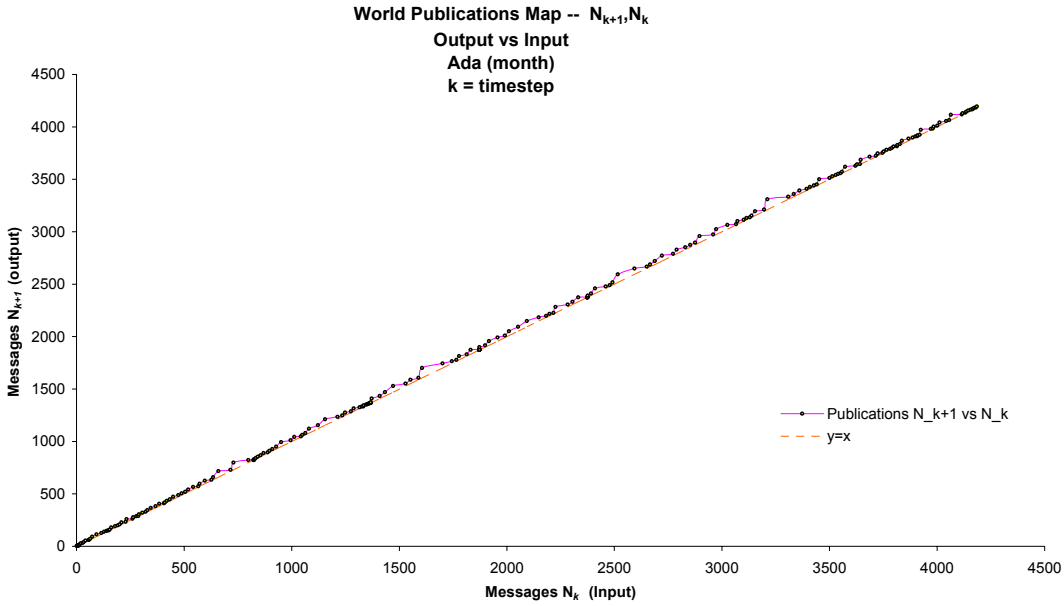
APPENDIX G – ACCESSION NUMBERS & DATES

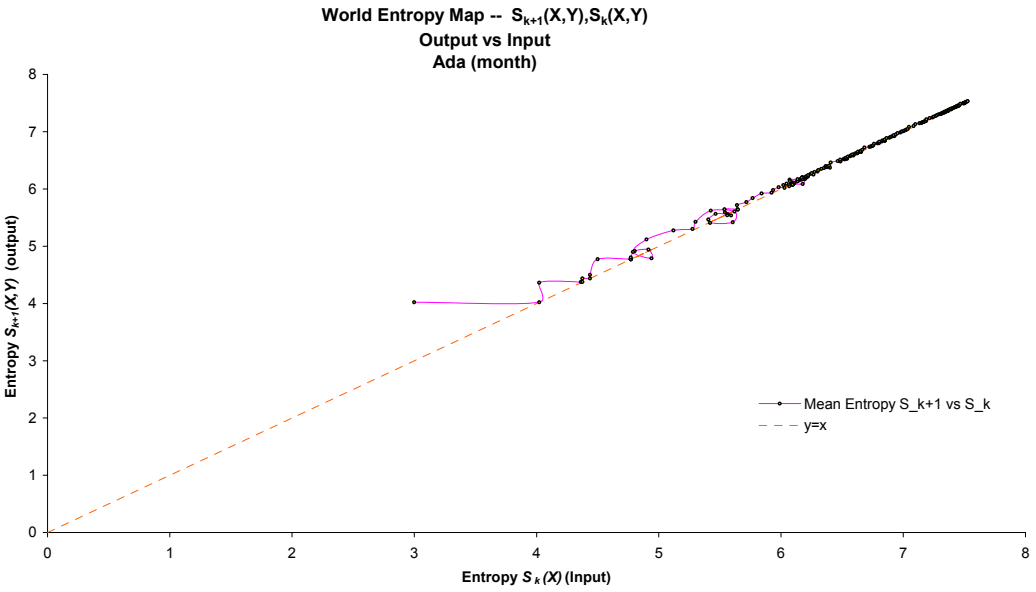
File 2 (1969 to present)

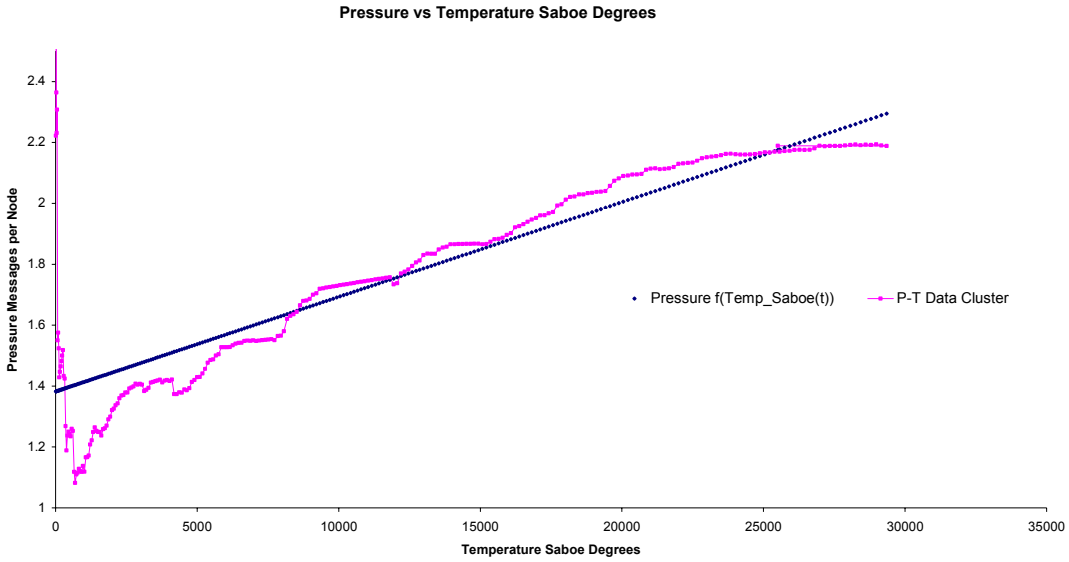
Year	Accession Numbers
1969	00000001-00077999
1970	00078000-00203427
1971	00203428-00329824
1972	00329825-00462218
1973	00462219-00581540
1974	00581541-00703915
1975	00703916-00839869
1976	00839870-00990949
1977	00990950-01126133
1978	01126134-01276268
1979	01276269-01432493
1980	01432494-01604845
1981	01604846-01772464
1982	01772465-01959518
1983	01959519-02153708
1984	02153709-02349062
1985	02349063-02556647
1986	02556648-02774876
1987	02774877-03021189
1988	03021190-03259711
1989	03259712-03506594
1990	03506595-03764942
1991	03764943-04024754
1992	04024755-04315714
1993	04315715-04559711
1994	04559712-04840653
1995	04840654-05145636
1996	05145637-05457178
1997	05457179-05787740
1998	05787741-06129398
1999	06129399-06483807
2000	06483808-06796501

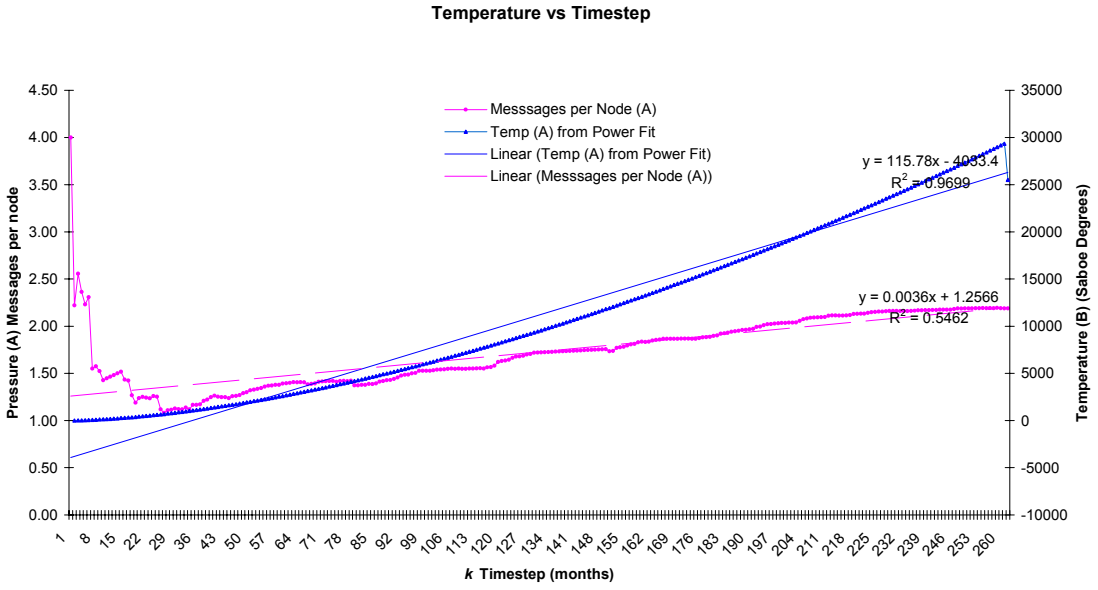
THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX H –AFFILIATION DISTRIBUTION MACRO OUTPUT CHARTS

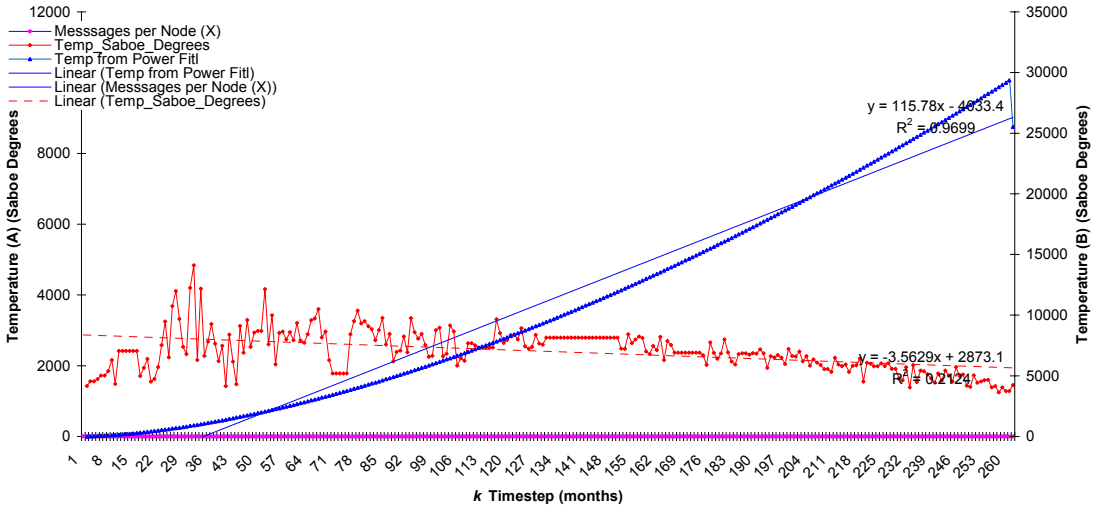




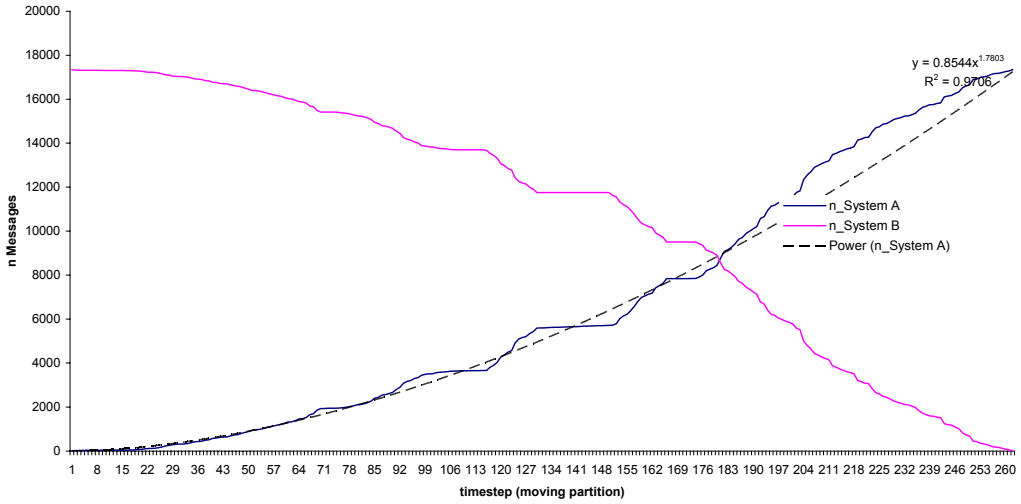

Pressure vs Temperature Saboe Degrees
Pressure Messages per Node
Temperature Saboe Degrees
Pressure f(Temp_Saboe(t))
P-T Data Cluster
2.4
2.2
2
1.8
1.6
1.4
1.2
1
0
5000
10000
15000
20000
25000
30000
35000

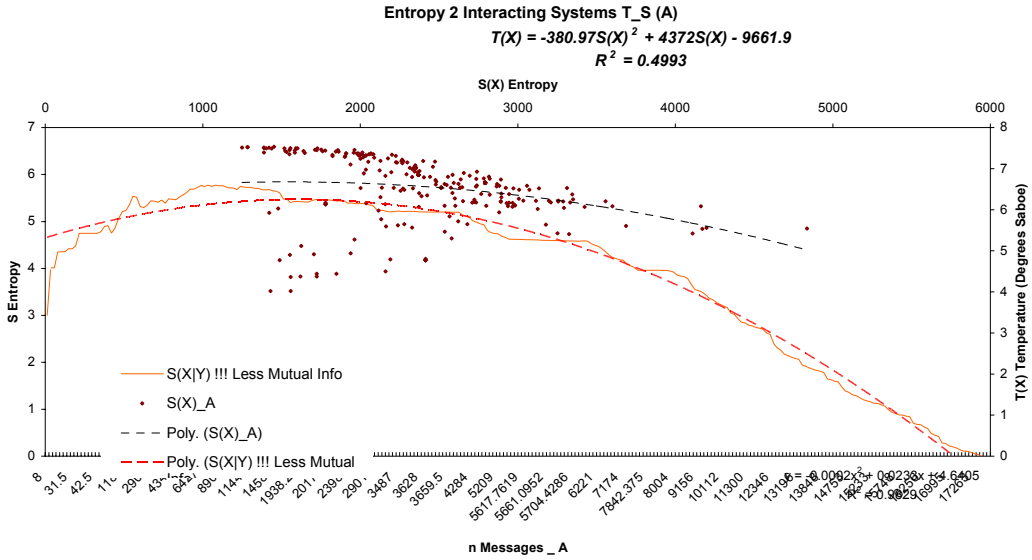

Temperature vs Timestep
Messsages per Node (A)
Temp (A) from Power Fit
Linear (Temp (A) from Power Fit)
Linear (Messsages per Node (A))
y = 115.78x - 4083.4
R2 = 0.9699
y = 0.0036x + 1.2566
R2 = 0.5462
Pressure (A) Messages per node
Temperature (B) (Saboe Degrees)
k Timestep (months)
4.50
4.00
3.50
3.00
2.50
2.00
1.50
1.00
0.50
0.00
35000
30000
25000
20000
15000
10000
5000
0
-5000
-10000
1
8
15
22
29
36
43
50
57
64
71
78
85
92
99
106
113
120
127
134
141
148
155
162
169
176
183
190
197
204
211
218
225
232
239
246
253
260

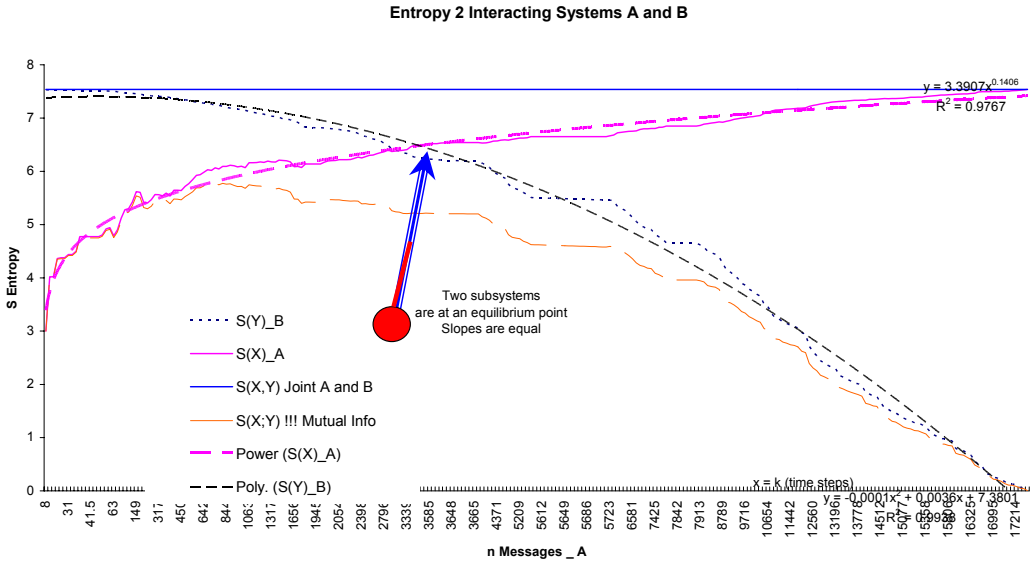
Temperature vs Timestep

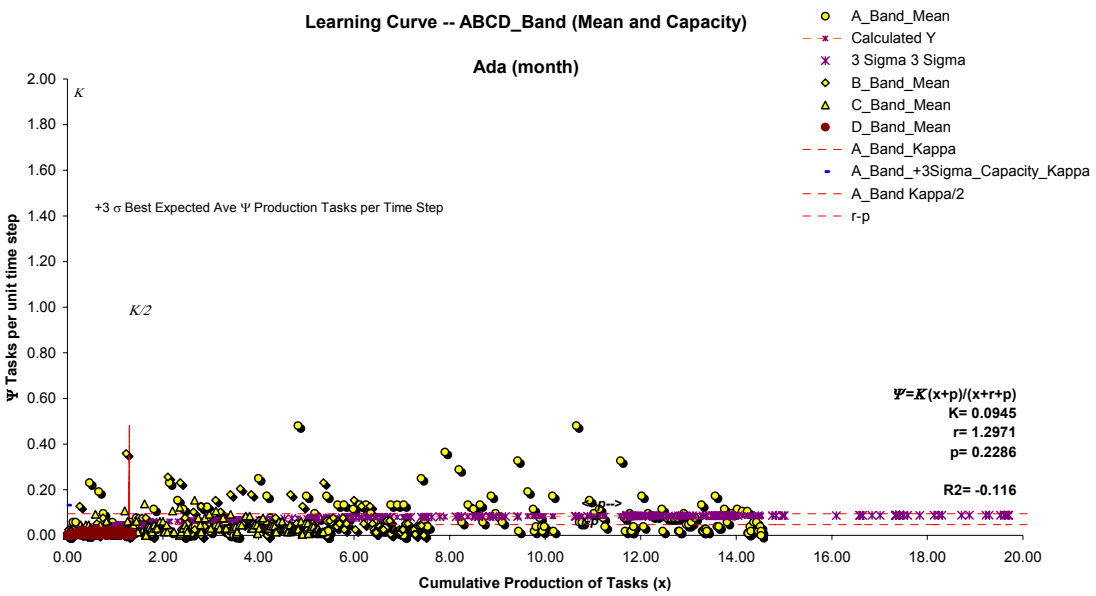


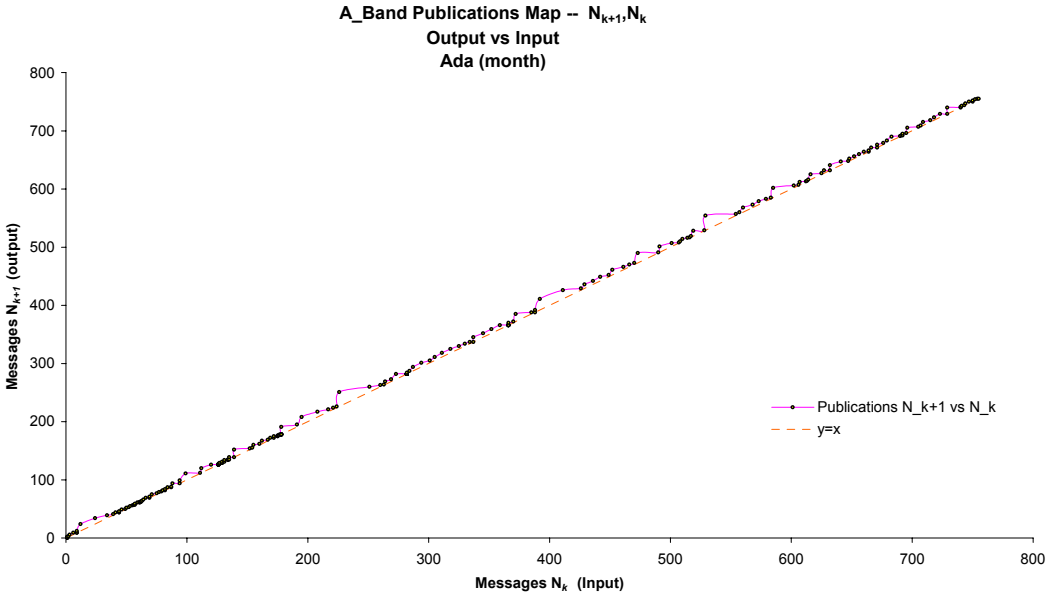
Interacting Systems A and B (Constant Messages in Total System AB)

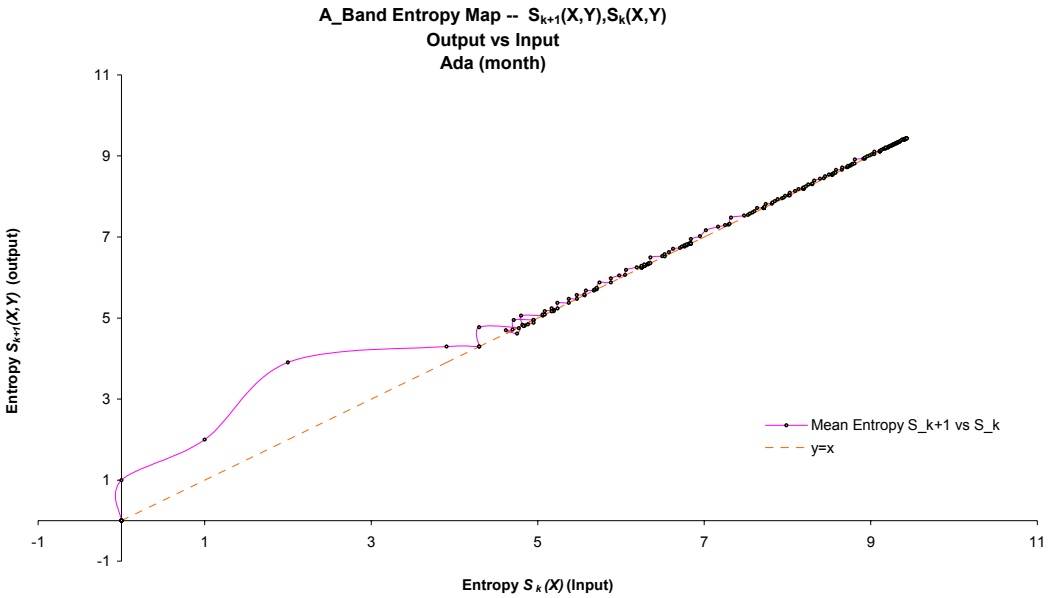


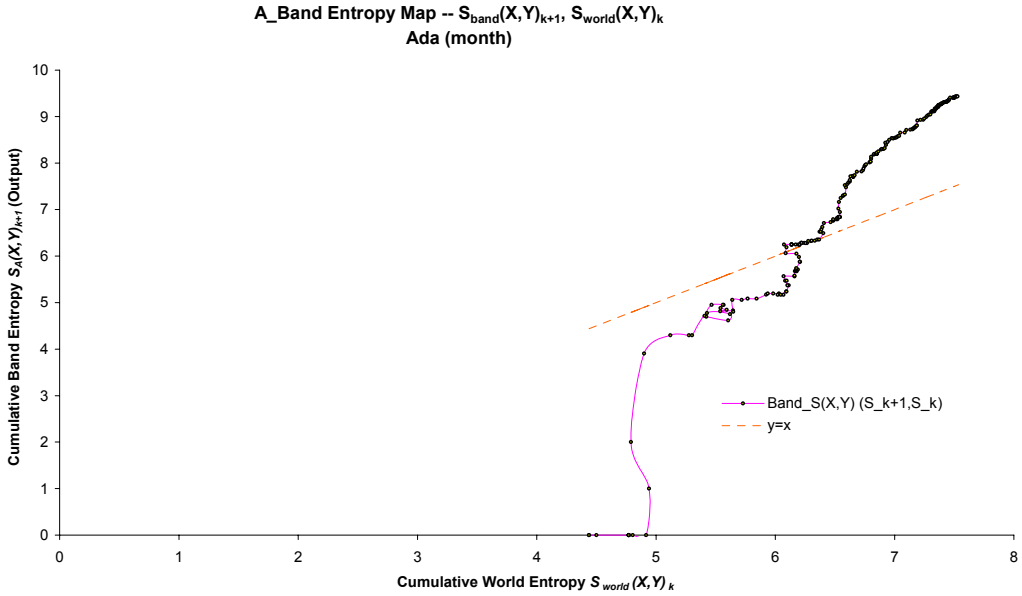

Entropy 2 Interacting Systems T_S (A)
T(X) = -380.97S(X)^2 + 4372S(X) - 9661.9
R^2 = 0.4993
S(X) Entropy
S Entropy
T(X) Temperature (Degrees Saboe)
S(X|Y) !!! Less Mutual Info
S(X)_A
Poly. (S(X)_A)
Poly. (S(X|Y) !!! Less Mutual
n Messages _ A

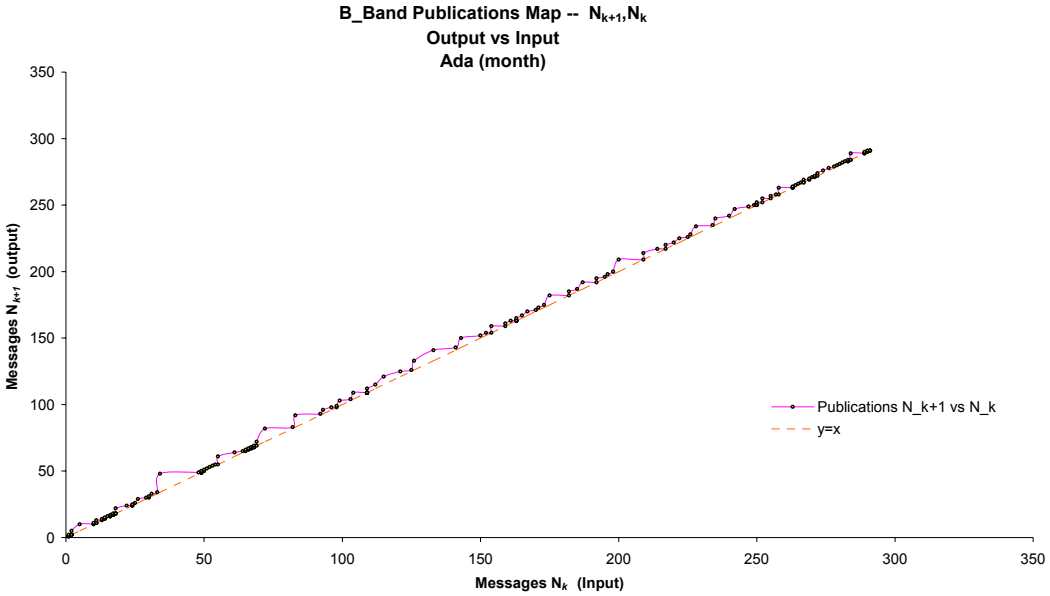

Entropy 2 Interacting Systems A and B
y = 3.3907x^0.1406
R^2 = 0.9767
S Entropy
S(Y)_B
S(X)_A
S(X,Y) Joint A and B
S(X;Y) !!! Mutual Info
Power (S(X)_A)
Poly. (S(Y)_B)
Two subsystems
are at an equilibrium point
Slopes are equal
x = k (time steps)
n Messages _ A

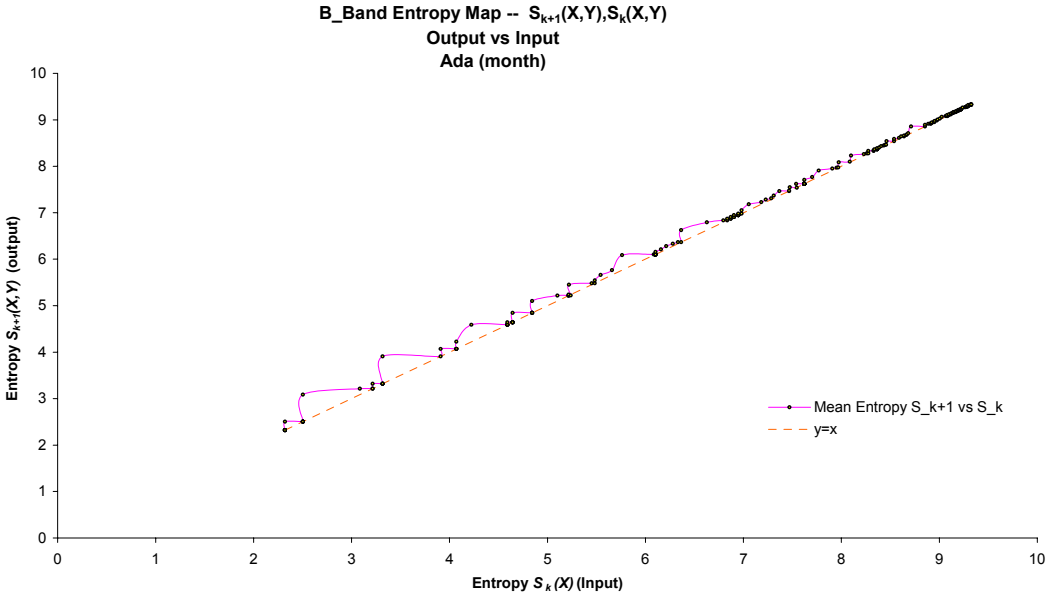

Learning Curve -- ABCD_Band (Mean and Capacity)
Ada (month)
A_Band_Mean
Calculated Y
3 Sigma 3 Sigma
B_Band_Mean
C_Band_Mean
D_Band_Mean
A_Band_Kappa
A_Band_+3Sigma_Capacity_Kappa
A_Band Kappa/2
r-p
K
+3 σ Best Expected Ave Ψ Production Tasks per Time Step
K/2
Ψ Tasks per unit time step
Ψ=K(x+p)/(x+r+p)
K= 0.0945
r= 1.2971
p= 0.2286
R2= -0.116
-p-->
Cumulative Production of Tasks (x)

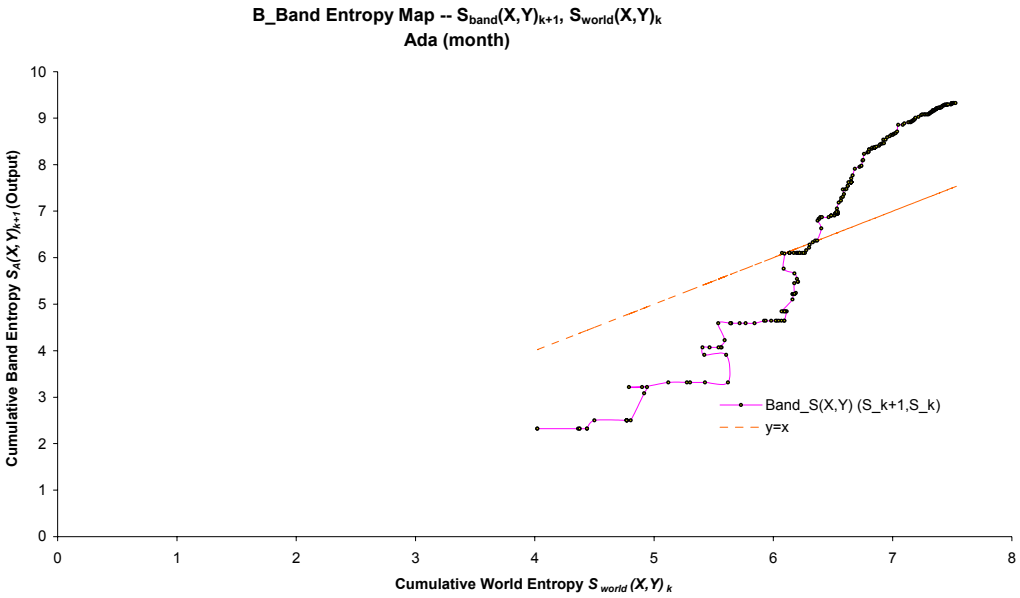

A_Band Publications Map -- N_{k+1},N_k
Output vs Input
Ada (month)
Messages N_{k+1} (output)
Messages N_k (Input)
Publications N_k+1 vs N_k
y=x

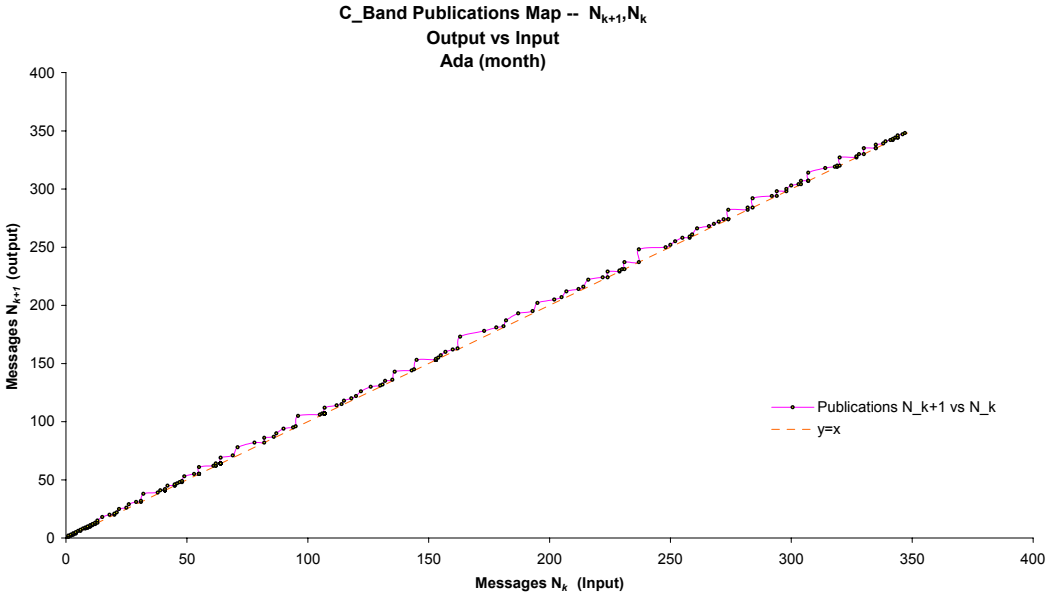

A_Band Entropy Map -- $S_{k+1}(X,Y),S_k(X,Y)$
Output vs Input
Ada (month)
Entropy $S_{k+1}(X,Y)$ (output)
Entropy $S_k(X)$ (Input)
Mean Entropy S_k+1 vs S_k
y=x
11
9
7
5
3
1
-1
-1
1
3
5
7
9
11

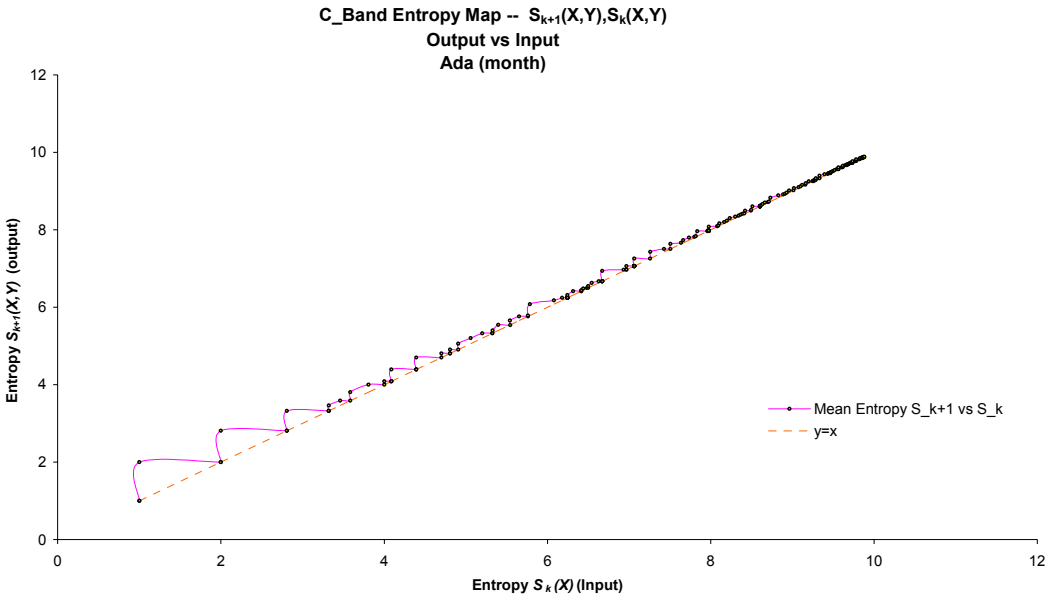

A_Band Entropy Map -- $S_{band}(X,Y)_{k+1}$, $S_{world}(X,Y)_k$
Ada (month)
Cumulative Band Entropy $S_A(X,Y)_{k+1}$ (Output)
Cumulative World Entropy $S_{world}(X,Y)_k$
Band_S(X,Y) (S_k+1,S_k)
y=x
10
9
8
7
6
5
4
3
2
1
0
0
1
2
3
4
5
6
7
8

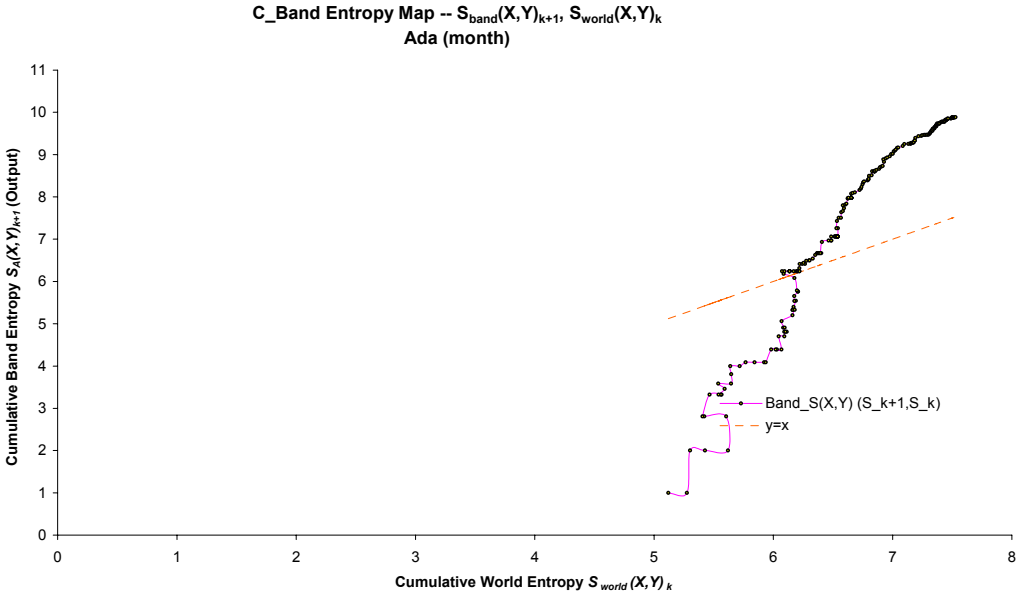

B_Band Publications Map -- N_{k+1},N_k
Output vs Input
Ada (month)
Messages N_{k+1} (output)
Messages N_k (Input)
Publications N_k+1 vs N_k
y=x
0
50
100
150
200
250
300
350

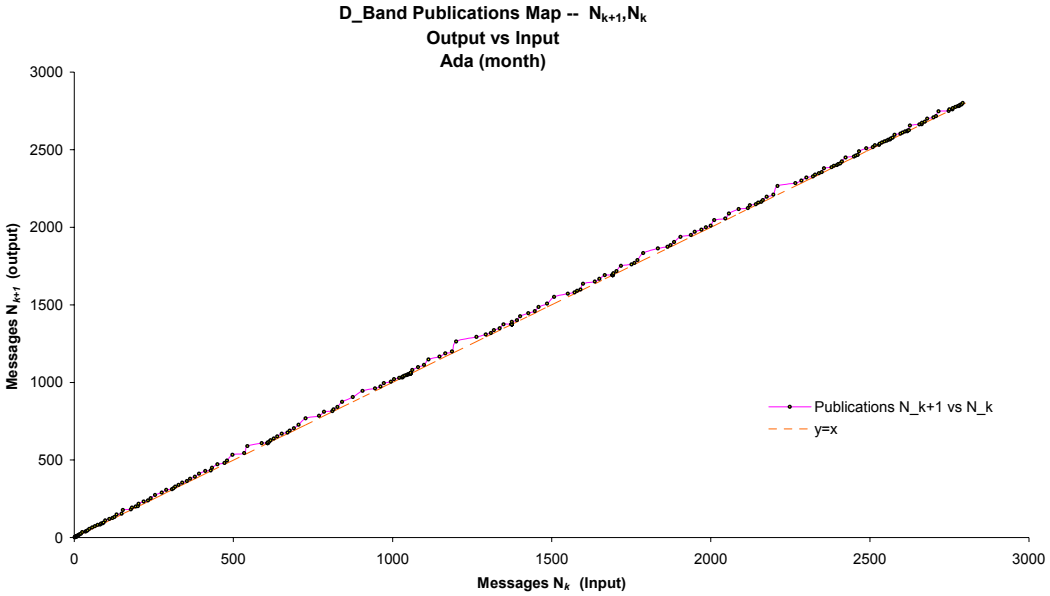

B_Band Entropy Map -- $S_{k+1}(X,Y)$,$S_k(X,Y)$
Output vs Input
Ada (month)
Entropy $S_{k+1}(X,Y)$ (output)
Entropy $S_k(X)$ (Input)
Mean Entropy S_k+1 vs S_k
y=x
0
1
2
3
4
5
6
7
8
9
10

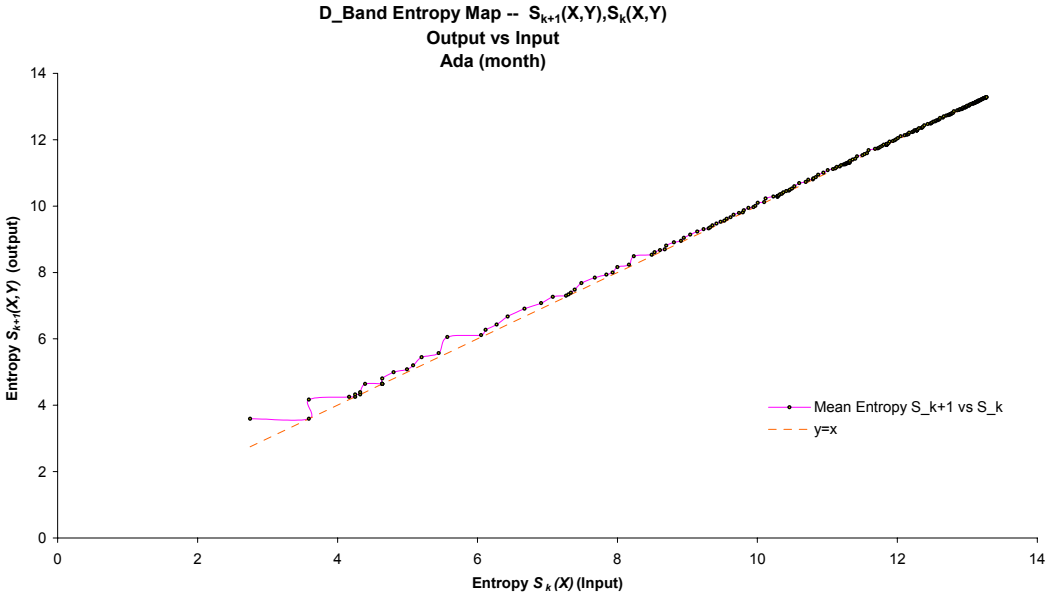

B_Band Entropy Map -- $S_{band}(X,Y)_{k+1}$, $S_{world}(X,Y)_k$
Ada (month)
Cumulative Band Entropy $S_A(X,Y)_{k+1}$ (Output)
Cumulative World Entropy $S_{world}(X,Y)_k$
Band_S(X,Y) (S_k+1,S_k)
y=x
0
1
2
3
4
5
6
7
8
9
10

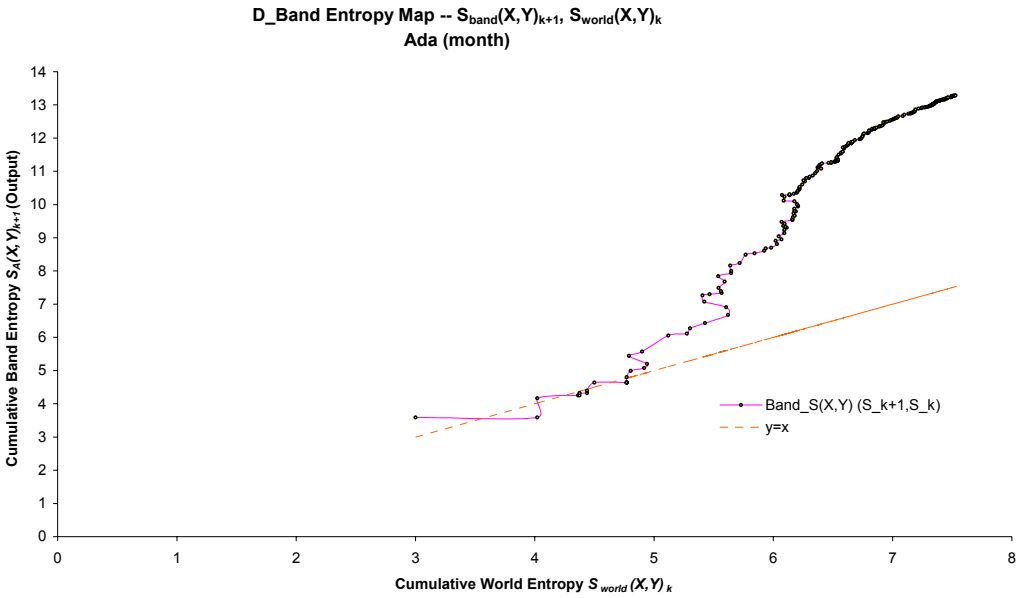

C_Band Publications Map -- N_{k+1},N_k
Output vs Input
Ada (month)
Messages N_{k+1} (output)
Messages N_k (Input)
Publications N_k+1 vs N_k
y=x
0
50
100
150
200
250
300
350
400

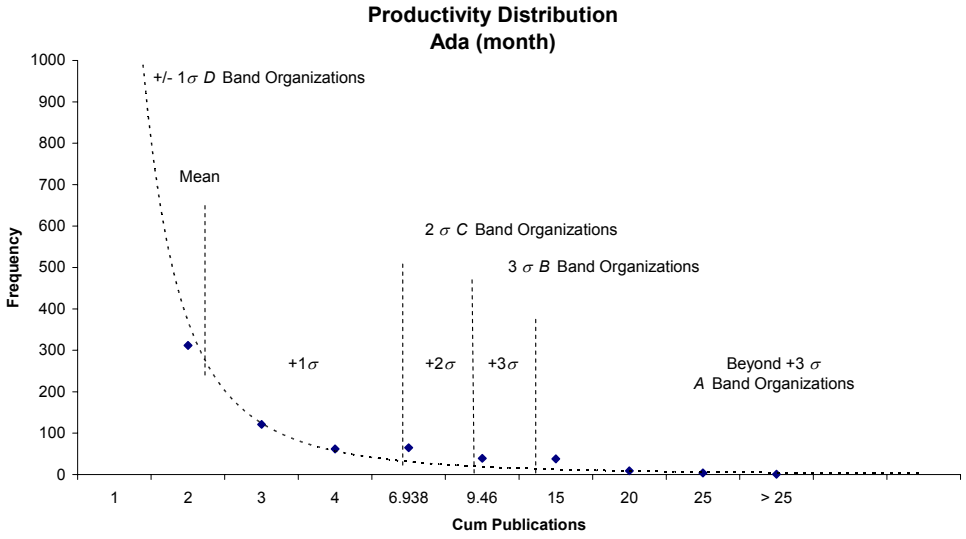

C_Band Entropy Map -- $S_{k+1}(X,Y),S_k(X,Y)$
Output vs Input
Ada (month)
Entropy $S_{k+1}(X,Y)$ (output)
Entropy $S_k(X)$ (Input)
Mean Entropy S_k+1 vs S_k
y=x

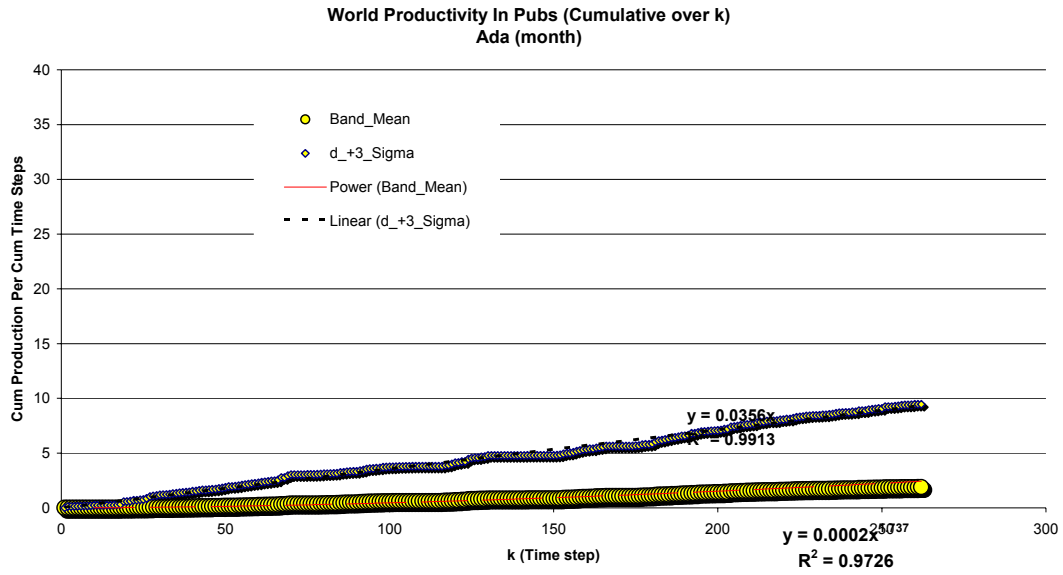

C_Band Entropy Map -- $S_{band}(X,Y)_{k+1}$, $S_{world}(X,Y)_k$
Ada (month)
Cumulative Band Entropy $S_A(X,Y)_{k+1}$ (Output)
Cumulative World Entropy $S_{world}(X,Y)_k$
Band_S(X,Y) (S_k+1,S_k)
y=x

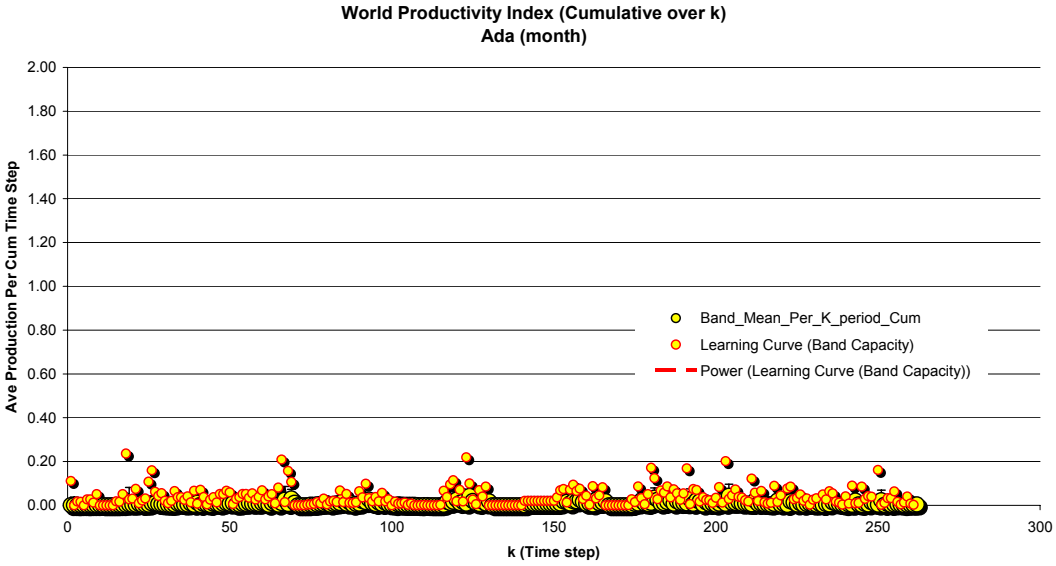

D_Band Publications Map -- N_{k+1},N_k
Output vs Input
Ada (month)
Messages N_{k+1} (output)
Messages N_k (Input)
Publications N_k+1 vs N_k
y=x
0
500
1000
1500
2000
2500
3000

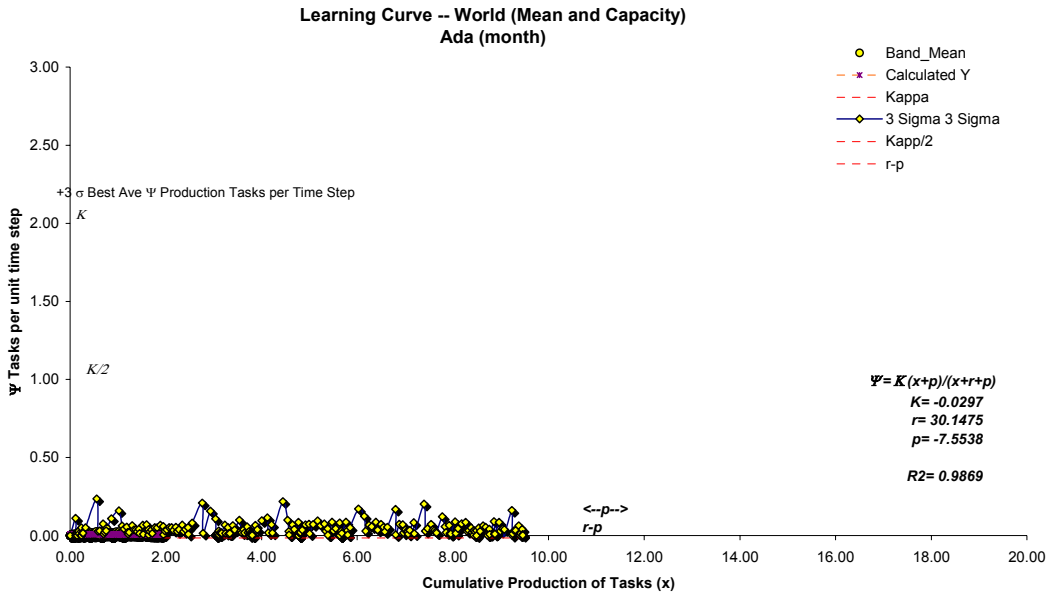

D_Band Entropy Map -- $S_{k+1}(X,Y)$,$S_k(X,Y)$
Output vs Input
Ada (month)
Entropy $S_{k+1}(X,Y)$ (output)
Entropy $S_k(X)$ (Input)
Mean Entropy S_k+1 vs S_k
y=x
0
2
4
6
8
10
12
14

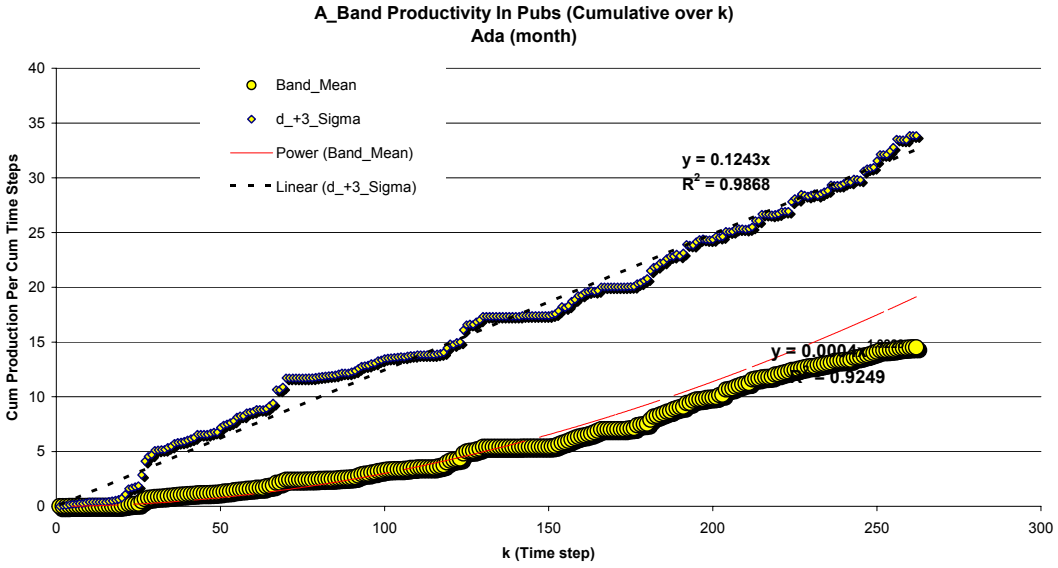

D_Band Entropy Map -- $S_{band}(X,Y)_{k+1}$, $S_{world}(X,Y)_k$
Ada (month)
Cumulative Band Entropy $S_A(X,Y)_{k+1}$ (Output)
Cumulative World Entropy $S_{world}(X,Y)_k$
Band_S(X,Y) (S_k+1,S_k)
y=x

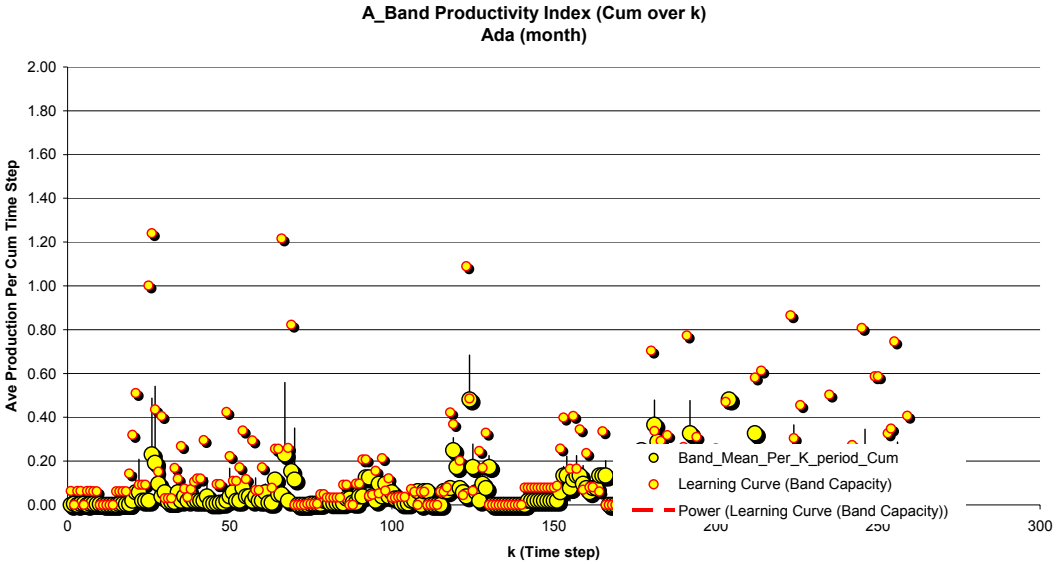

Productivity Distribution
Ada (month)
Frequency
+/- 1σ D Band Organizations
Mean
2 σ C Band Organizations
3 σ B Band Organizations
+1σ
+2σ
+3σ
Beyond +3 σ
A Band Organizations
Cum Publications

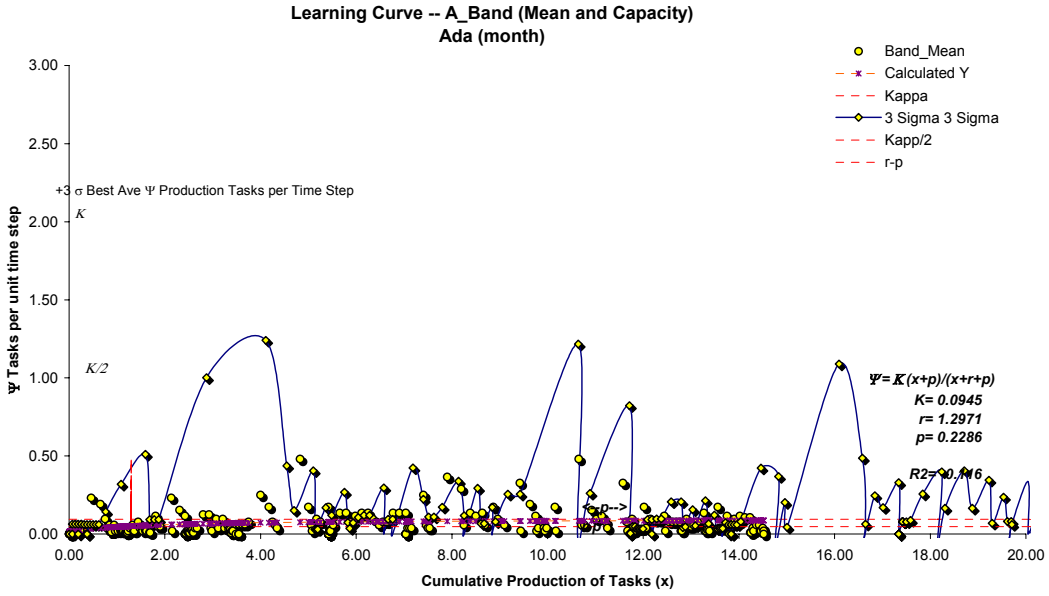

World Productivity In Pubs (Cumulative over k)
Ada (month)
Cum Production Per Cum Time Steps
k (Time step)
Band_Mean
d_+3_Sigma
Power (Band_Mean)
Linear (d_+3_Sigma)
y = 0.0356x
R² = 0.9913
y = 0.0002x
R² = 0.9726

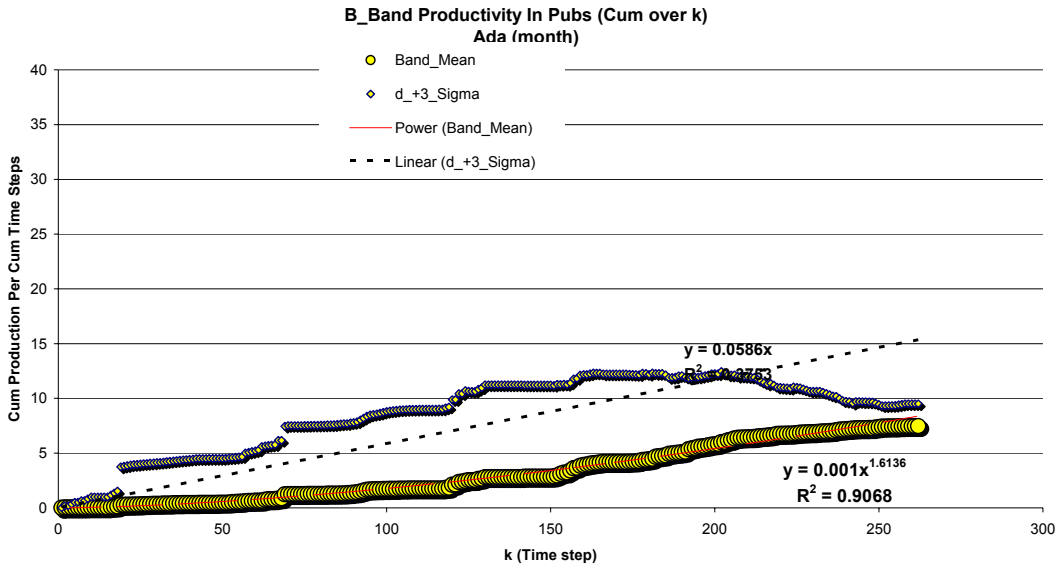

World Productivity Index (Cumulative over k)
Ada (month)
Ave Production Per Cum Time Step
k (Time step)
Band_Mean_Per_K_period_Cum
Learning Curve (Band Capacity)
Power (Learning Curve (Band Capacity))

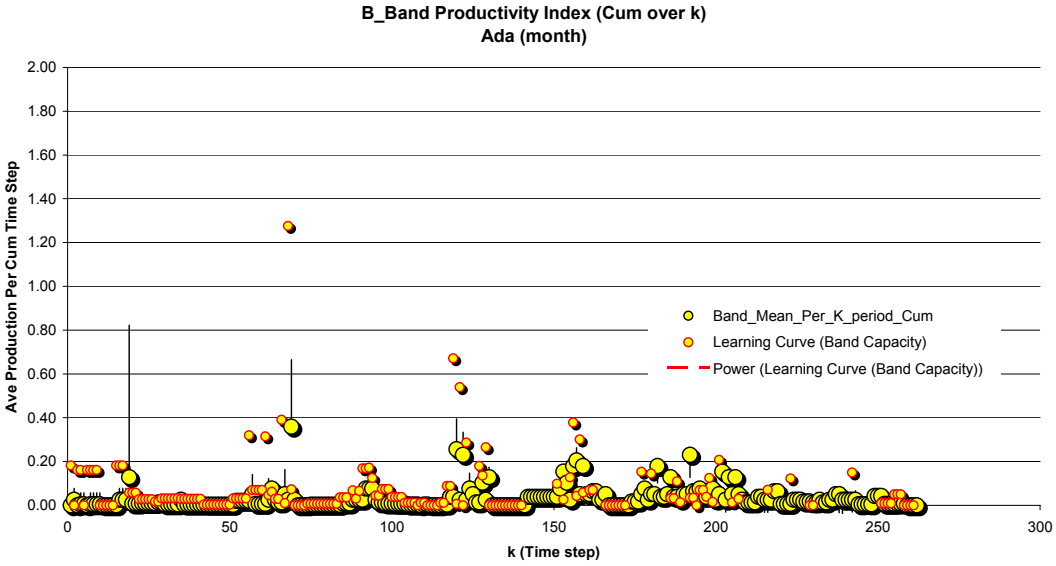

Learning Curve -- World (Mean and Capacity)
Ada (month)
Band_Mean
Calculated Y
Kappa
3 Sigma 3 Sigma
Kapp/2
r-p
+3 σ Best Ave Ψ Production Tasks per Time Step
K
K/2
Ψ Tasks per unit time step
Ψ= K (x+p)/(x+r+p)
K= -0.0297
r= 30.1475
p= -7.5538
R2= 0.9869
<--p-->
r-p
Cumulative Production of Tasks (x)

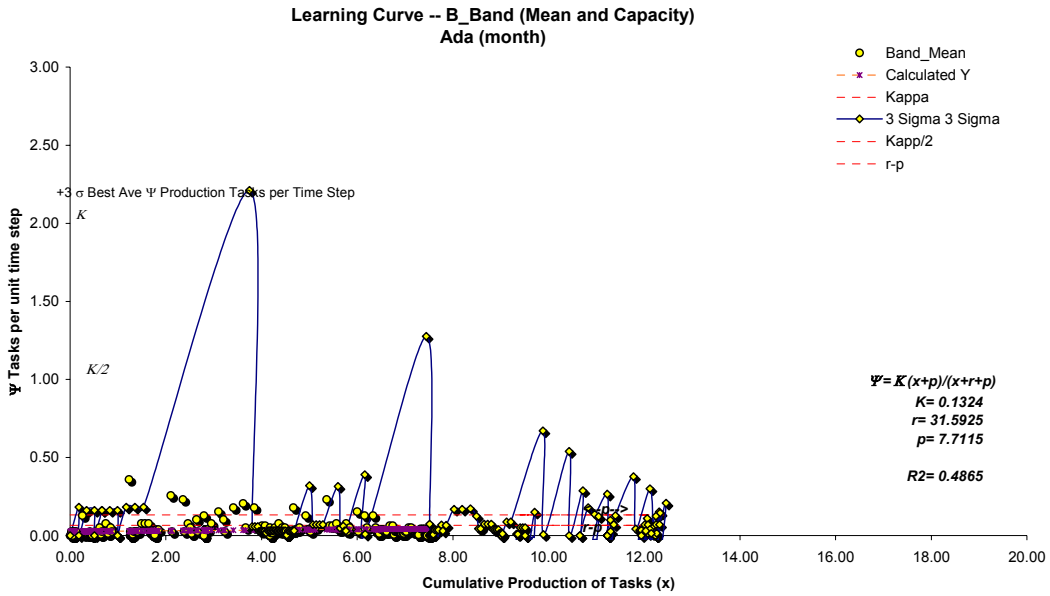

A_Band Productivity In Pubs (Cumulative over k)
Ada (month)
Band_Mean
d_+3_Sigma
Power (Band_Mean)
Linear (d_+3_Sigma)
y = 0.1243x
R2 = 0.9868
y = 0.0004x
= 0.9249
Cum Production Per Cum Time Steps
k (Time step)

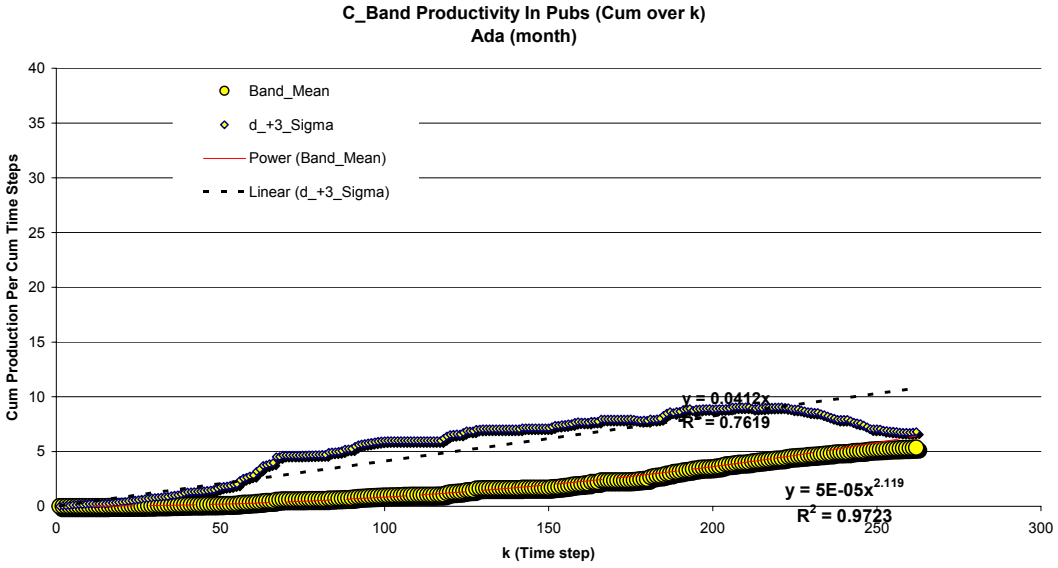

A_Band Productivity Index (Cum over k)
Ada (month)
Ave Production Per Cum Time Step
2.00
1.80
1.60
1.40
1.20
1.00
0.80
0.60
0.40
0.20
0.00
0
50
100
150
200
250
300
k (Time step)
Band_Mean_Per_K_period_Cum
Learning Curve (Band Capacity)
Power (Learning Curve (Band Capacity))

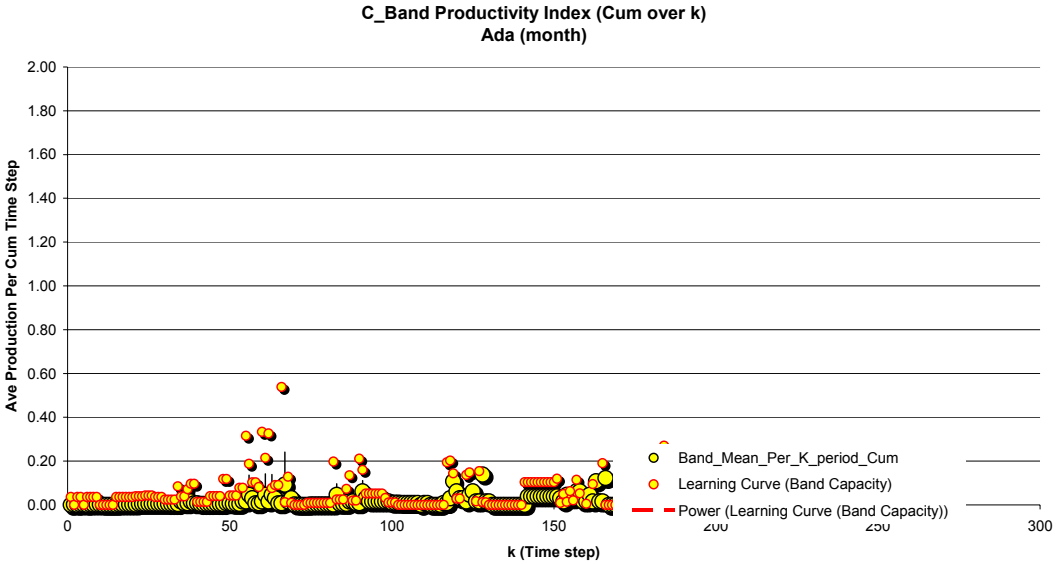

Learning Curve -- A_Band (Mean and Capacity)
Ada (month)
Band_Mean
Calculated Y
Kappa
3 Sigma 3 Sigma
Kapp/2
r-p
3.00
2.50
2.00
1.50
1.00
0.50
0.00
+3 σ Best Ave Ψ Production Tasks per Time Step
K
K/2
Ψ Tasks per unit time step
Ψ= K(x+p)/(x+r+p)
K= 0.0945
r= 1.2971
p= 0.2286
R2= 0.7146
r-p-->
-p
0.00
2.00
4.00
6.00
8.00
10.00
12.00
14.00
16.00
18.00
20.00
Cumulative Production of Tasks (x)

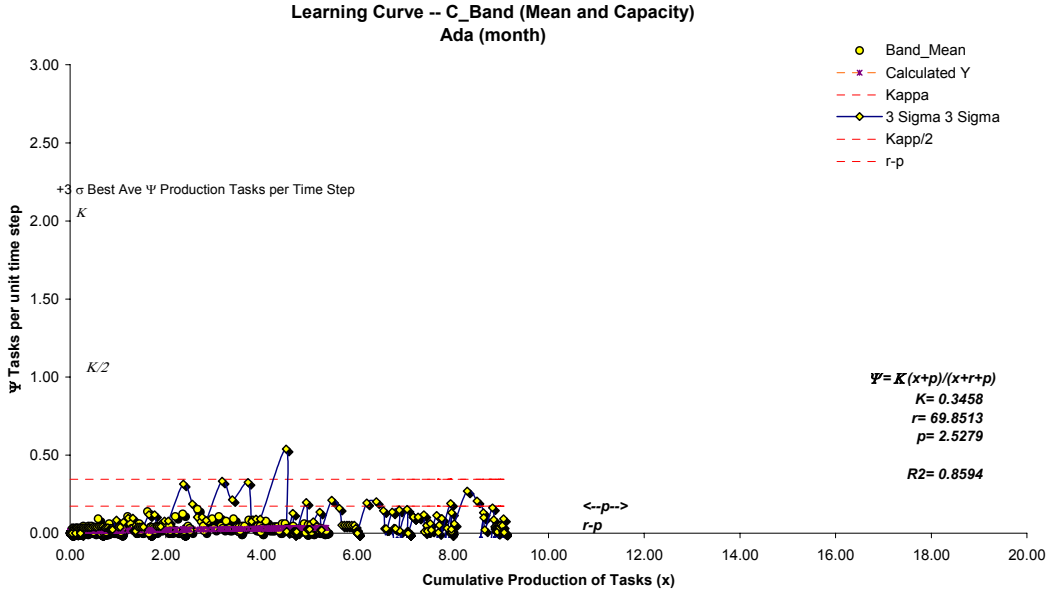

B_Band Productivity In Pubs (Cum over k)
Ada (month)
Band_Mean
d_+3_Sigma
Power (Band_Mean)
Linear (d_+3_Sigma)
Cum Production Per Cum Time Steps
k (Time step)
y = 0.0586x
y = 0.001x^1.6136
R^2 = 0.9068

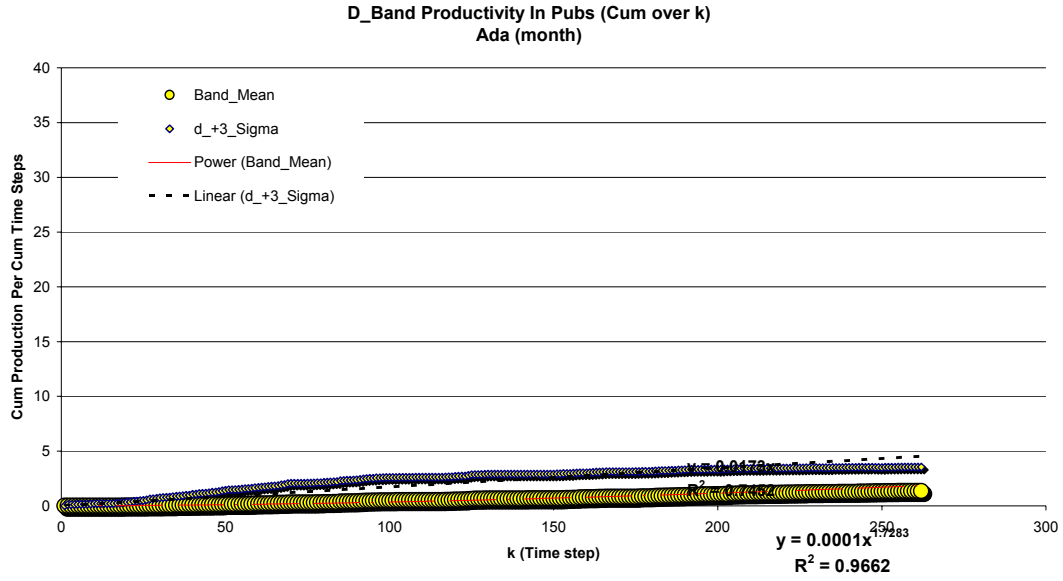

B_Band Productivity Index (Cum over k)
Ada (month)
Band_Mean_Per_K_period_Cum
Learning Curve (Band Capacity)
Power (Learning Curve (Band Capacity))
Ave Production Per Cum Time Step
k (Time step)

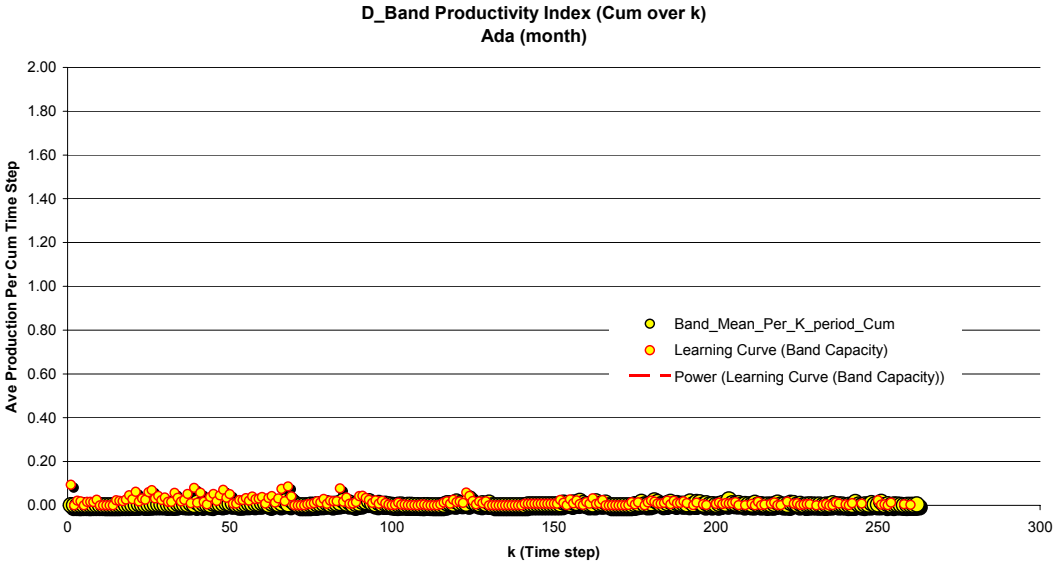

Learning Curve -- B_Band (Mean and Capacity)
Ada (month)
Band_Mean
Calculated Y
Kappa
3 Sigma 3 Sigma
Kapp/2
r-p
+3 σ Best Ave Ψ Production Tasks per Time Step
K
K/2
Ψ= K (x+p)/(x+r+p)
K= 0.1324
r= 31.5925
p= 7.7115
R2= 0.4865
Ψ Tasks per unit time step
Cumulative Production of Tasks (x)
3.00
2.50
2.00
1.50
1.00
0.50
0.00
0.00
2.00
4.00
6.00
8.00
10.00
12.00
14.00
16.00
18.00
20.00

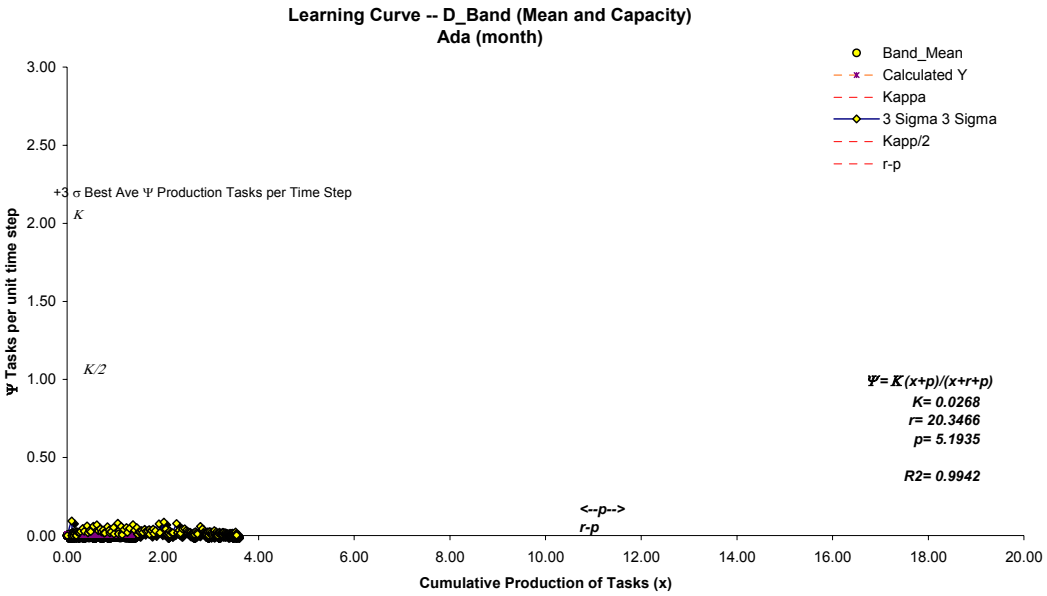

C_Band Productivity In Pubs (Cum over k)
Ada (month)
Band_Mean
d_+3_Sigma
Power (Band_Mean)
Linear (d_+3_Sigma)
y = 0.0412x
R² = 0.7619
y = 5E-05x^2.119
R² = 0.9723
Cum Production Per Cum Time Steps
k (Time step)
40
35
30
25
20
15
10
5
0
0
50
100
150
200
250
300


C_Band Productivity Index (Cum over k)
Ada (month)
Ave Production Per Cum Time Step
2.00
1.80
1.60
1.40
1.20
1.00
0.80
0.60
0.40
0.20
0.00
0
50
100
150
200
250
300
k (Time step)
Band_Mean_Per_K_period_Cum
Learning Curve (Band Capacity)
Power (Learning Curve (Band Capacity))


Learning Curve -- C_Band (Mean and Capacity)
Ada (month)
Band_Mean
Calculated Y
Kappa
3 Sigma 3 Sigma
Kapp/2
r-p
+3 σ Best Ave Ψ Production Tasks per Time Step
K
K/2
Ψ Tasks per unit time step
3.00
2.50
2.00
1.50
1.00
0.50
0.00
0.00
2.00
4.00
6.00
8.00
10.00
12.00
14.00
16.00
18.00
20.00
Cumulative Production of Tasks (x)
<--p-->
r-p
Ψ= K(x+p)/(x+r+p)
K= 0.3458
r= 69.8513
p= 2.5279
R2= 0.8594


D_Band Productivity In Pubs (Cum over k)
Ada (month)
Band_Mean
d_+3_Sigma
Power (Band_Mean)
Linear (d_+3_Sigma)
Cum Production Per Cum Time Steps
k (Time step)
y = 0.0001x^1.7283
R² = 0.9662


D_Band Productivity Index (Cum over k)
Ada (month)
Band_Mean_Per_K_period_Cum
Learning Curve (Band Capacity)
Power (Learning Curve (Band Capacity))
Ave Production Per Cum Time Step
k (Time step)


Learning Curve -- D_Band (Mean and Capacity)
Ada (month)
Band_Mean
Calculated Y
Kappa
3 Sigma 3 Sigma
Kapp/2
r-p
+3 σ Best Ave Ψ Production Tasks per Time Step
K
K/2
Ψ= K (x+p)/(x+r+p)
K= 0.0268
r= 20.3466
p= 5.1935
R2= 0.9942
<--p-->
r-p
Ψ Tasks per unit time step
Cumulative Production of Tasks (x)
3.00
2.50
2.00
1.50
1.00
0.50
0.00
0.00
2.00
4.00
6.00
8.00
10.00
12.00
14.00
16.00
18.00
20.00

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX I – ENTROPY LAMBDA MACRO OUTPUT CHARTS

Entropy Lamba Macro Outputs

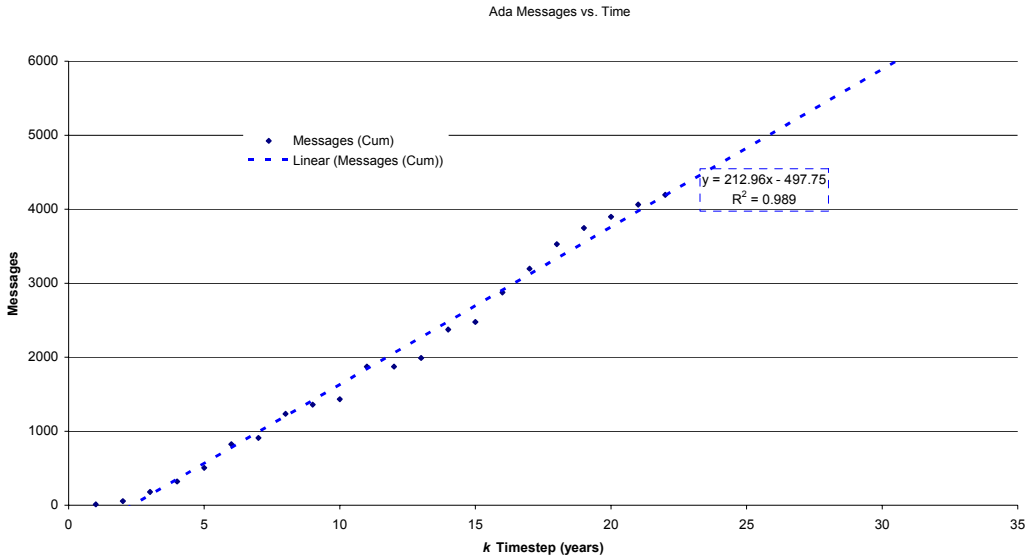
Correlate_S_Lambda sheet:

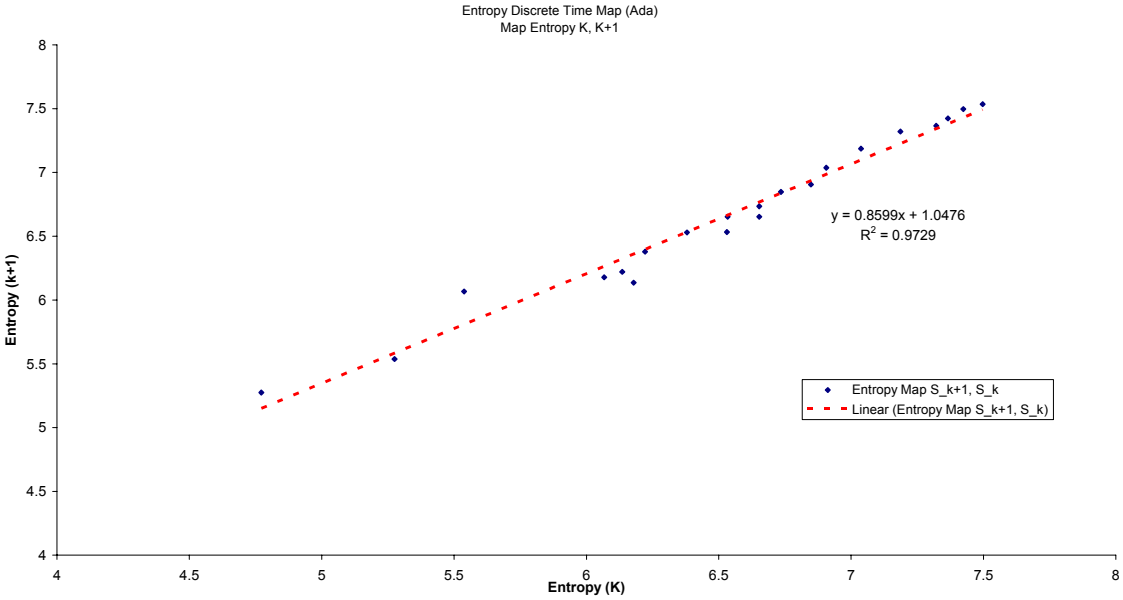
m_s	0.141	m_lambda	0.001	
b_s	4.7841	b_lambda	0.995	beta / authors

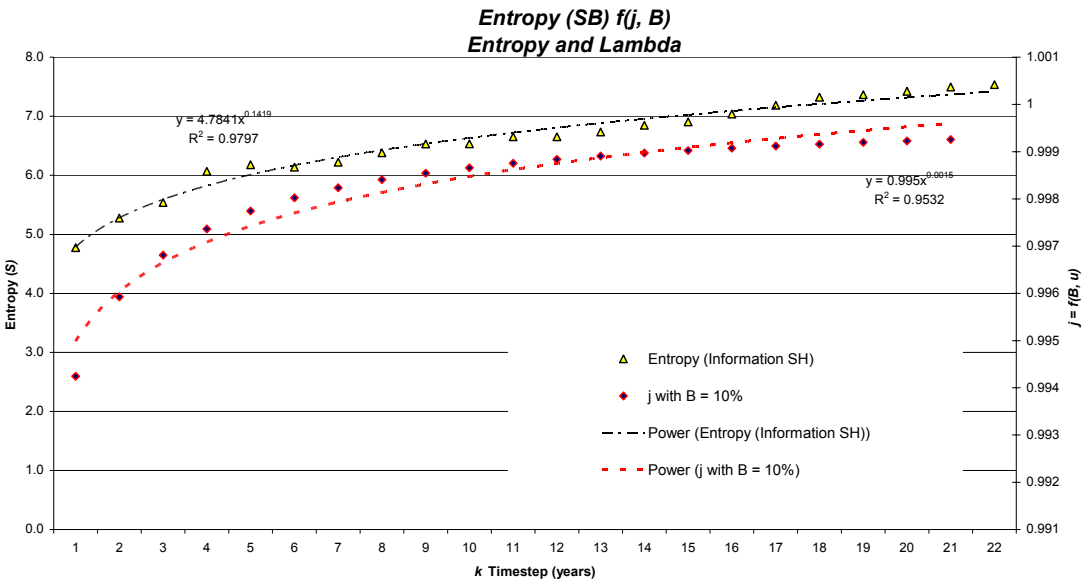
TimeStep (k)	month/year	Entropy (S)	Lambda	Entropy(Lambda_	delta lambda	Lambda from beta goal	Beta goal to min R^2	Mindshare
2	12/1/1980	5.275314635	0.994245629	5.251483194	0.000567938	0.995657995	0.132905382	0.001399004
3	12/1/1981	5.538028259	0.995924291	5.517431891	0.00042421	0.996006903	0.10208582	0.000302925
4	12/1/1982	6.067531972	0.996810999	6.098500836	0.000959071	0.996714464	0.097042997	0.000167028
5	12/1/1983	6.178802542	0.997364327	6.198915371	0.000404526	0.996829915	0.083052859	9.66855E-05
6	12/1/1984	6.135650861	0.99774469	6.162750561	0.000734394	0.99678855	0.0700925	5.11251E-05
7	12/1/1985	6.221374187	0.998023296	6.252729979	0.000983186	0.996891026	0.063436288	4.06904E-05
8	12/1/1986	6.379739802	0.998236756	6.399690706	0.000398039	0.99705529	0.059736593	2.80322E-05
9	12/1/1987	6.530833163	0.998405885	6.553265561	0.000503212	0.997222992	0.057268005	2.44317E-05
10	12/1/1988	6.533517799	0.998543422	6.560591028	0.00073296	0.997230893	0.052462711	2.10356E-05
11	12/1/1989	6.652679089	0.998657616	6.67988532	0.000740179	0.99735835	0.050683887	1.55807E-05
12	12/1/1990	6.652679089	0.998754051	6.680920936	0.000797602	0.997359446	0.047053378	1.44646E-05
13	12/1/1991	6.734791162	0.998836647	6.762053036	0.00074321	0.997444832	0.045402533	1.31259E-05
14	12/1/1992	6.848164916	0.99890824	6.874037342	0.000669382	0.997561031	0.044642434	1.06241E-05
15	12/1/1993	6.906260151	0.998970934	6.930567623	0.000590853	0.997618977	0.043102516	9.80717E-06
16	12/1/1994	7.037288817	0.999026323	7.060339643	0.000531341	0.997750243	0.04316863	8.39857E-06
17	12/1/1995	7.186357395	0.99907564	7.208449878	0.000488078	0.997897162	0.043854029	7.59903E-06
18	12/1/1996	7.321492306	0.999119851	7.342667513	0.000448389	0.998027734	0.044528711	6.91547E-06
19	12/1/1997	7.366025475	0.999159727	7.386051569	0.000401044	0.998069433	0.043429703	6.29325E-06
20	12/1/1998	7.423671507	0.999195888	7.454260879	0.00093571	0.998134504	0.0430129	5.94183E-06
21	12/1/1999	7.496781098	0.999228842	7.525898257	0.000847809	0.998202213	0.042806722	5.62432E-06
22	12/1/2000	7.535631281	0.999259005	7.562510818	0.00072251	0.99823657	0.041934104	5.29204E-06

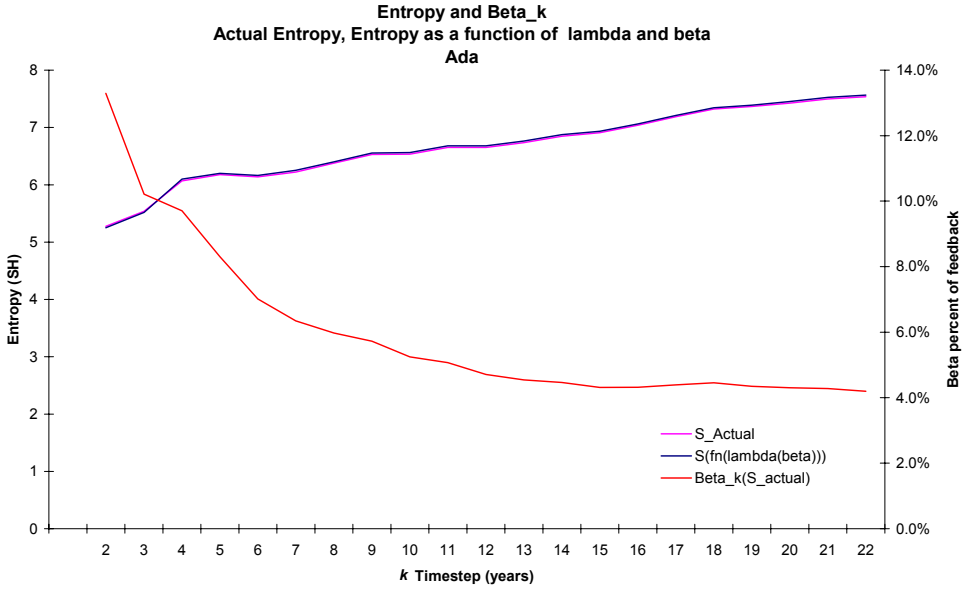
Entropy Lambda sheet:

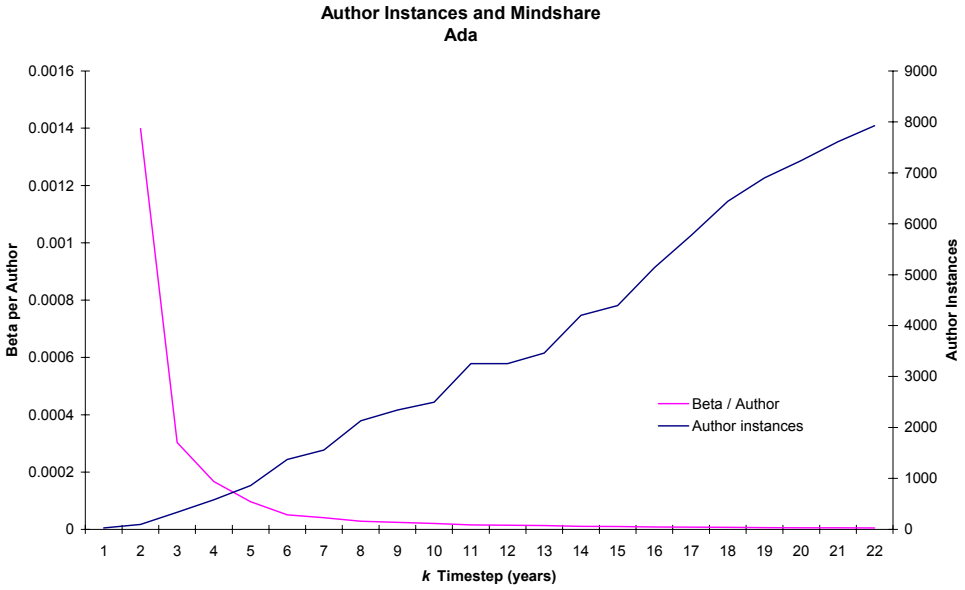
					y = 212.96x - 497.75 R2 = 0.989	y = 4.7841x0.1419 R2 = 0.9797	y = 0.995x0.0015 R2 = 0.9532		
TimeStep (k)	month/year	Records (cumu	Entropy (S)	S(k+1)	du_(t-c)	du_(t)	B_y_10%	Lambda_B10%	10%
1	12/1/1979	11	4.771928095	5.275314635					
2	12/1/1980	56	5.275314635	5.538028259	212.96	0.371907208	4.281069006	0.994245629	0.1
3	12/1/1981	179	5.538028259	6.067531972	212.96	0.262525929	4.542103968	0.995924291	0.1
4	12/1/1982	321	6.067531972	6.178802542	212.96	0.205045305	5.070720974	0.996810999	0.1
5	12/1/1983	505	6.178802542	6.135650861	212.96	0.169279407	5.181438215	0.997364327	0.1
6	12/1/1984	824	6.135650861	6.221374187	212.96	0.144739623	5.137906171	0.99774469	0.1
7	12/1/1985	910	6.221374187	6.379739802	212.96	0.126788582	5.223350891	0.998023296	0.1
8	12/1/1986	1235	6.379739802	6.530833163	212.96	0.113048567	5.381503046	0.998236756	0.1
9	12/1/1987	1361	6.530833163	6.533517799	212.96	0.102170389	5.532427278	0.998405885	0.1
10	12/1/1988	1434	6.533517799	6.652679089	212.96	0.093329591	5.534974377	0.998543422	0.1
11	12/1/1989	1873	6.652679089	6.652679089	212.96	0.085992989	5.654021473	0.998657616	0.1
12	12/1/1990	1873	6.652679089	6.734791162	212.96	0.07979996	5.653925038	0.998754051	0.1
13	12/1/1991	1991	6.734791162	6.848164916	212.96	0.074497555	5.735954515	0.998836647	0.1
14	12/1/1992	2374	6.848164916	6.906260151	212.96	0.06990293	5.849256676	0.99890824	0.1
15	12/1/1993	2478	6.906260151	7.037288817	212.96	0.065880513	5.907289217	0.998970934	0.1
16	12/1/1994	2875	7.037288817	7.186357395	212.96	0.062327585	6.038262494	0.999026323	0.1
17	12/1/1995	3196	7.186357395	7.321492306	212.96	0.059164846	6.187281755	0.99907564	0.1
18	12/1/1996	3529	7.321492306	7.366025475	212.96	0.056330069	6.322372455	0.999119851	0.1
19	12/1/1997	3746	7.366025475	7.423671507	212.96	0.053773712	6.366865748	0.999159727	0.1
20	12/1/1998	3899	7.423671507	7.496781098	212.96	0.051455831	6.424475619	0.999195888	0.1
21	12/1/1999	4064	7.496781098	7.535631281	212.96	0.049343845	6.497552256	0.999228842	0.1
22	12/1/2000	4195	7.535631281	0	212.96	0.047410909	6.536372275	0.999259005	0.1


Ada Messages vs. Time
Messages (Cum)
Linear (Messages (Cum))
y = 212.96x - 497.75
R2 = 0.989
Messages
k Timestep (years)
0
1000
2000
3000
4000
5000
6000
0
5
10
15
20
25
30
35


Entropy Discrete Time Map (Ada)
Map Entropy K, K+1
y = 0.8599x + 1.0476
R2 = 0.9729
Entropy Map S_k+1, S_k
Linear (Entropy Map S_k+1, S_k)
Entropy (k+1)
Entropy (K)
4
4.5
5
5.5
6
6.5
7
7.5
8


Entropy (SB) f(j, B)
Entropy and Lambda
y = 4.7841x^0.1419
R² = 0.9797
y = 0.995x^0.0015
R² = 0.9532
Entropy (S)
j = f(B, u)
k Timestep (years)
Entropy (Information SH)
j with B = 10%
Power (Entropy (Information SH))
Power (j with B = 10%)


Entropy and Beta_k
Actual Entropy, Entropy as a function of lambda and beta
Ada
Entropy (SH)
Beta percent of feedback
k Timestep (years)
S_Actual
S(fn(lambda(beta)))
Beta_k(S_actual)

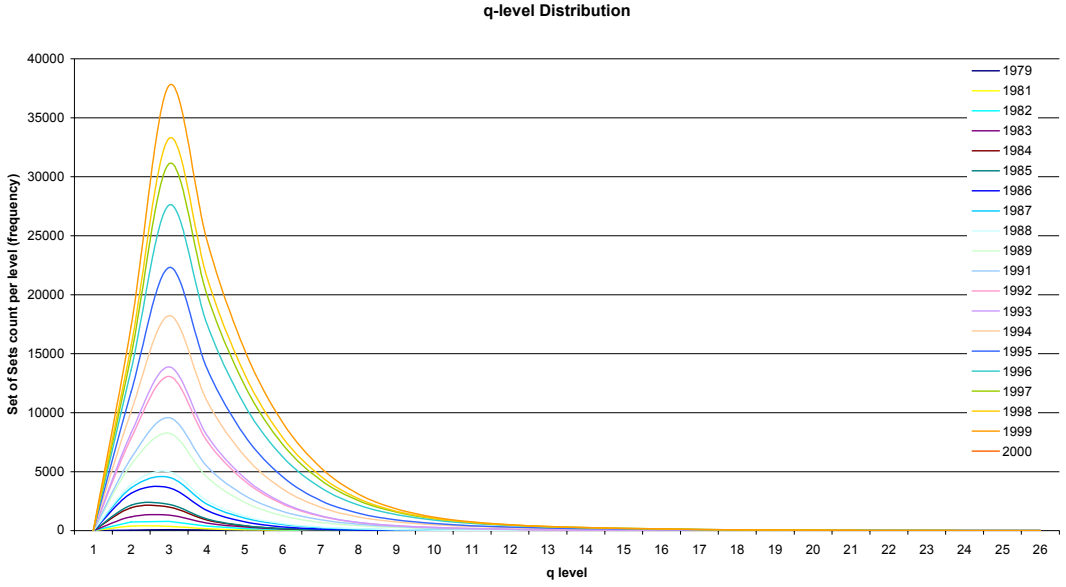

Author Instances and Mindshare
Ada
Beta per Author
Author Instances
0.0016
0.0014
0.0012
0.001
0.0008
0.0006
0.0004
0.0002
0
9000
8000
7000
6000
5000
4000
3000
2000
1000
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
k Timestep (years)
Beta / Author
Author instances

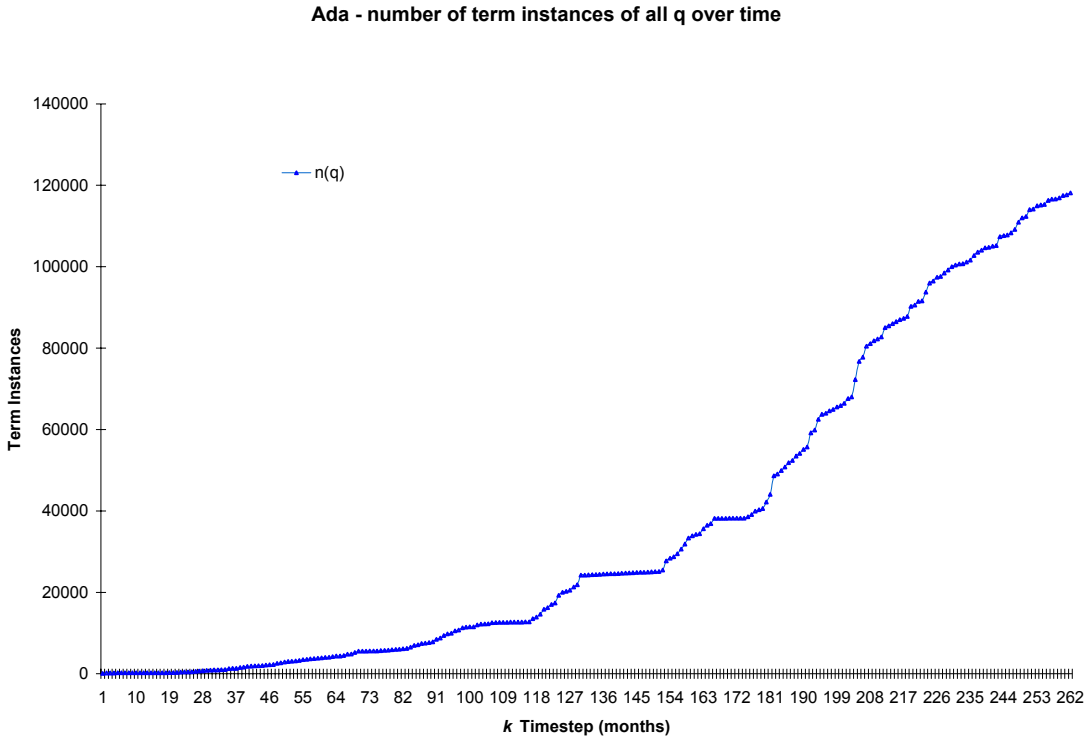
APPENDIX J – Q LEVEL ANALYSIS MACRO OUTPUT CHARTS

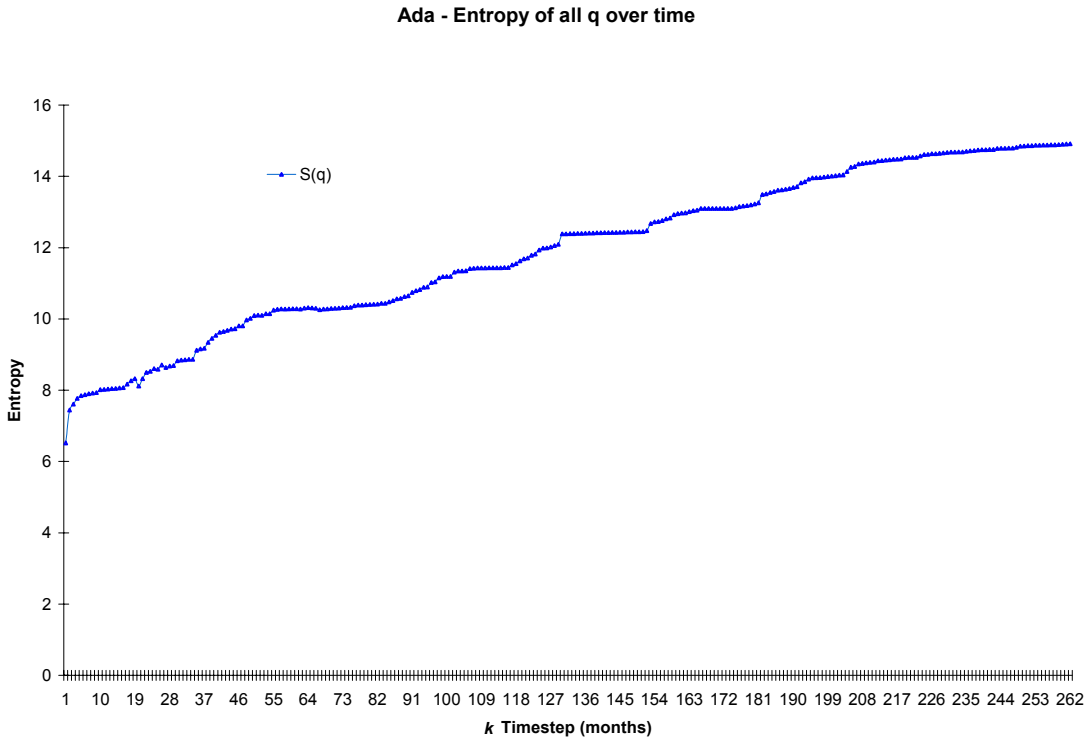
ada_qlevels_term_month2.xls

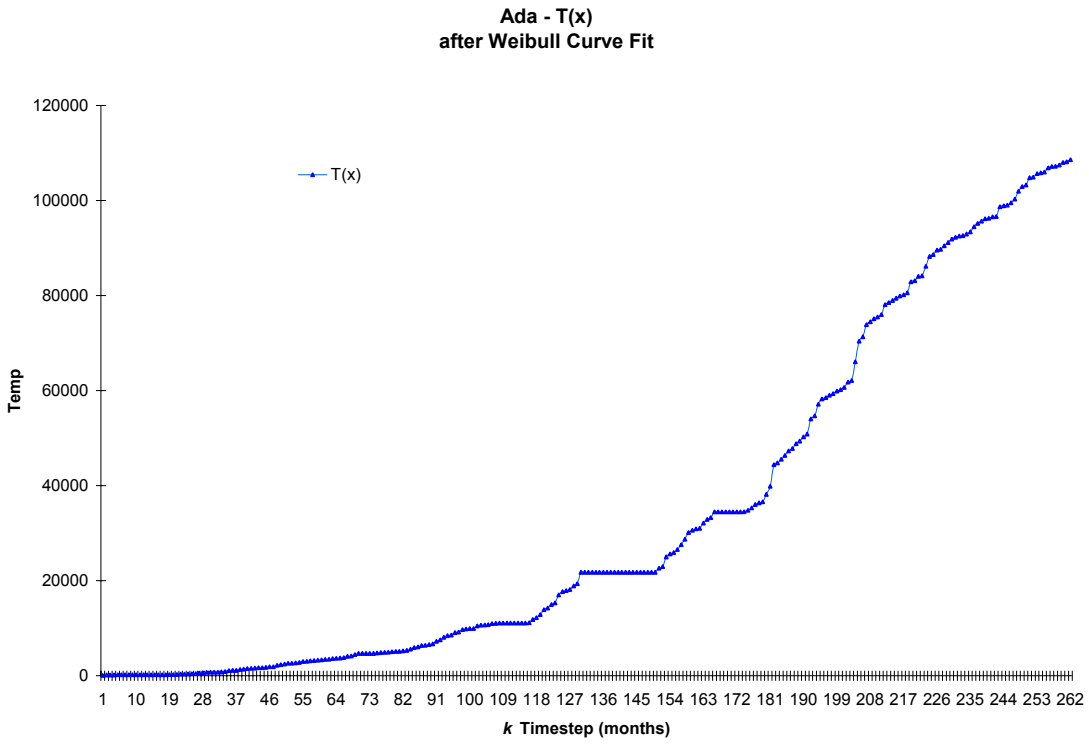
	A	B	C	D	E	F	G	H	I	J	K	L
1												
2						Number of term instances in each q-Level per year						
3	year		1	2	3	4	5	6	7	8	9	10
4	1979	0	40	86	57	35	20	10	4	1	0	0
5	1980	0	118	141	79	44	23	10	4	1	0	0
6	1981	0	377	365	167	81	39	16	5	1	0	0
7	1982	0	731	791	381	188	93	43	18	7	2	0
8	1983	0	1184	1321	642	314	155	75	35	16	6	1
9	1984	0	1935	2009	898	392	175	77	35	16	6	1
10	1985	0	2150	2237	997	430	186	79	35	16	6	1
11	1986	0	3165	3637	1707	748	324	136	58	27	12	4
12	1987	0	3628	4521	2254	1077	521	261	139	79	43	21
13	1988	0	3903	5005	2536	1230	601	305	163	94	53	27
14	1989	0	5593	8257	4537	2398	1274	710	425	281	199	143
15	1990	0	5593	8257	4537	2398	1274	710	425	281	199	143
16	1991	0	6141	9588	5438	2981	1636	926	549	349	234	158
17	1992	0	7839	13067	7602	4206	2265	1223	687	420	271	176
18	1993	0	8278	13881	8081	4453	2374	1263	701	424	271	176
19	1994	0	10112	18229	10992	6311	3527	1981	1161	742	509	364
20	1995	0	11761	22307	13742	8048	4554	2557	1476	914	604	417
21	1996	0	13618	27600	17511	10625	6257	3657	2174	1360	898	615
22	1997	0	14862	31098	19982	12286	7314	4298	2546	1573	1019	684
23	1998	0	15689	33252	21462	13243	7893	4621	2715	1655	1060	703
24	1999	0	16654	35907	23317	14463	8640	5043	2937	1764	1108	720
25	2000	0	17347	37737	24585	15298	9159	5347	3105	1852	1153	743
26												

25_Term_summary | q_summary_year | q_summary_monthly_instance




Ada - number of term instances of all q over time
n(q)
Term Instances
140000
120000
100000
80000
60000
40000
20000
0
1 10 19 28 37 46 55 64 73 82 91 100 109 118 127 136 145 154 163 172 181 190 199 208 217 226 235 244 253 262
k Timestep (months)


Ada - Entropy of all q over time
S(q)
Entropy
16
14
12
10
8
6
4
2
0
1 10 19 28 37 46 55 64 73 82 91 100 109 118 127 136 145 154 163 172 181 190 199 208 217 226 235 244 253 262
k Timestep (months)


Ada - T(x)
after Weibull Curve Fit
T(x)
Temp
120000
100000
80000
60000
40000
20000
0
1 10 19 28 37 46 55 64 73 82 91 100 109 118 127 136 145 154 163 172 181 190 199 208 217 226 235 244 253 262
k Timestep (months)

APPENDIX K – TECH OASIS EXPORT SCRIPT

```
'*************************************************************************
' Macro: dataExport.tmf
' Author: Matt Behnke
'
' Description: This macro exports the data fields from Tech OASIS into MS Excel
' along with a unique identifier so the data can be imported in to MS Access
' where relationships can be maintained to allow further data manipulation
'
' *********NEXT VERSION---> Export directly to MS Access.
'
'*************************************************************************

Option Explicit

Dim nStatus, i, j
Dim exApp, strUniqueIDField
Dim strActiveDS, nRows, nCols
Dim exRowCount, strUniqueValue, strView, strVal, NameLength
Dim n, dataField1, dataField2, dataField3, dataField4, dataField5
Dim nDataFields, booleanIncremented

nStatus = App.GetActiveDSName(strActiveDS)

'get the field of the unique identifier

msgbox("Select unique identifier field  ")
nStatus = Dataset.PromptForField(strUniqueIDField)

'************************
'*Next version, ask the user the number of
'*datafields to export...
'************************


'prompt the user to select the datafields to export

 msgbox("Select data field 1 ")
 nStatus = Dataset.PromptForField(dataField1)
 msgbox("Select data field 2 ")
 nStatus = Dataset.PromptForField(dataField2)
 msgbox("Select data field 3 ")
 nStatus = Dataset.PromptForField(dataField3)
 msgbox("Select data field 4 ")
 nStatus = Dataset.PromptForField(dataField4)
' msgbox("Select data field 5 ")
' nStatus = Dataset.PromptForField(dataField5)


'Open Excel Workbook
Set exApp = CreateObject("Excel.Application")
exApp.Visible = True
exApp.Workbooks.Add
```

```
getData(dataField1)
getData(dataField2)
getData(dataField3)
getData(dataField4)
'getData(dataField5)
'
'-------------------------------------------------------------------------one
Function getData(datafield)

  'create a matrix in Tech oasis of the datafield on the left and the unique ID field
  'on the top.. Syntax:
     'View.CreateMatrix(leftfieldname, leftcontent, topfieldname, topcontent,
     '                  matrixtype, returnviewname)

  nStatus =
View.CreateMatrix(dataField,"UNGROUPED",strUniqueIDField,"UNGROUPED","COOCCURENCE",st
rView)

  'sort the matrix sort by the unique ID field.
  nStatus=Matrix.Sort("ROW",2,"DESCEND")

  'obtain the number of rows and columns in the created matrix
  nstatus=Matrix.GetNumColumns(nCols)
  nstatus=Matrix.GetNumRows(nRows)

  'create a new excel datasheet
  exApp.Sheets.Add
  exApp.Sheets(1).Name = datafield

  'put in column headers
  nStatus=Matrix.GetValue(2,0,strVal) 'start of columns (unique ID field)
  exApp.cells(1, 1).formula = strVal
  nStatus=Matrix.GetValue(0,2,strVal) 'start ofrows (datafield name)
  exApp.cells(1, 2).formula = strVal


  exRowCount = 2 'counter for the number of rows in the excel sheet

      for i = 3 To nCols
            nStatus=Matrix.GetValue(2,i,strUniqueValue)
            exApp.cells(exRowCount, 1).formula  = strUniqueValue

            for j = 3 to nRows
                  nstatus=Matrix.GetValue(j,i,strVal)

                     NameLength = InStr(strVal, "")
                  if NameLength <> 0 then  'if string isn't null then add values to sheet
                        nStatus=Matrix.GetValue(j,2,strVal)
                        exApp.cells(exRowCount, 2).formula  = strVal
                        exApp.cells(exRowCount, 1).formula  = strUniqueValue
                        exRowCount = exRowCount + 1
                        booleanIncremented = true
                  end if
            next 'row
            if booleanIncremented = false then
```

```
                exRowCount = exRowCount + 1
            end if
        next 'next column
end function
```

APPENDIX L – ASSIGN ACCESSION NUMBER MACRO

```
'************************
'  Macro: AssignAN
'  Description: Assigns the date and year to the Title worksheet that has been
'               exported from Tech OASIS. Works with INSPEC database data between
'               Jan 1969 and Dec 2000.
'
'  Future work: remove the need to copy the "AN" worksheet to the workbook that
'               contains the "Title" worksheet.
'************************


Sub assignAN()

nRowsAN = CountRows("AN", 1)
nRowsTitle = CountRows("Title", 1)

Sheets("Title").Select
Sheets("Title").Cells(1, 3) = "PubDate"
Sheets("Title").Cells(1, 4) = "PubYear"

   For x = 2 To nRowsTitle
      compare = Sheets("Title").Cells(x, 1)
      For i = 2 To nRowsAN
         timeInt = Sheets("AN").Cells(i, 1)
         startAN = Sheets("AN").Cells(i, 2)
         endAN = Sheets("AN").Cells(i, 3)
         If compare >= startAN And compare <= endAN Then
            Sheets("Title").Cells(x, 3) = timeInt
            Sheets("Title").Cells(x, 4) = Year(timeInt)
         End If
      Next i
   Next x

End Sub

Function CountRows(ByVal sheetName As String, ByVal colNum As Integer) As Double
   On Error Resume Next
   Dim currCell As Range, rowNum As Double

   Sheets("" & sheetName).Select

   If IsNumeric(colNum) Then
   Else
      colNum = 1
   End If

   rowNum = 1
   Set currCell = ActiveSheet.Cells(rowNum, colNum)
   Do While currCell.Value <> ""
      rowNum = rowNum + 1
      Set currCell = ActiveSheet.Cells(rowNum, colNum)
   Loop
```

```
CountRows = rowNum - 1
```

End Function

The following table maps the accession numbers to the month and year.

TimeInt	Start	End
Jan-69	1	6,501
Feb-69	6,502	13,001
Mar-69	13,002	19,501
Apr-69	19,502	26,000
May-69	26,001	32,500
Jun-69	32,501	39,000
Jul-69	39,001	45,500
Aug-69	45,501	52,000
Sep-69	52,001	58,500
Oct-69	58,501	64,999
Nov-69	65,000	71,499
Dec-69	71,500	77,999
Jan-70	78,000	88,452
Feb-70	88,453	98,905
Mar-70	98,906	109,357
Apr-70	109,358	119,809
May-70	119,810	130,261
Jun-70	130,262	140,714
Jul-70	140,715	151,166
Aug-70	151,167	161,618
Sep-70	161,619	172,070
Oct-70	172,071	182,523
Nov-70	182,524	192,975
Dec-70	192,976	203,427
Jan-71	203,428	213,961
Feb-71	213,962	224,494
Mar-71	224,495	235,027
Apr-71	235,028	245,560
May-71	245,561	256,093
Jun-71	256,094	266,626
Jul-71	266,627	277,159
Aug-71	277,160	287,692
Sep-71	287,693	298,225
Oct-71	298,226	308,758
Nov-71	308,759	319,291
Dec-71	319,292	329,824
Jan-72	329,825	340,858
Feb-72	340,859	351,891
Mar-72	351,892	362,923
Apr-72	362,924	373,956
May-72	373,957	384,989
Jun-72	384,990	396,022

Jul-72	396,023	407,054
Aug-72	407,055	418,087
Sep-72	418,088	429,120
Oct-72	429,121	440,153
Nov-72	440,154	451,185
Dec-72	451,186	462,218
Jan-73	462,219	472,162
Feb-73	472,163	482,106
Mar-73	482,107	492,049
Apr-73	492,050	501,993
May-73	501,994	511,936
Jun-73	511,937	521,880
Jul-73	521,881	531,823
Aug-73	531,824	541,766
Sep-73	541,767	551,710
Oct-73	551,711	561,653
Nov-73	561,654	571,597
Dec-73	571,598	581,540
Jan-74	581,541	591,739
Feb-74	591,740	601,937
Mar-74	601,938	612,135
Apr-74	612,136	622,332
May-74	622,333	632,530
Jun-74	632,531	642,728
Jul-74	642,729	652,926
Aug-74	652,927	663,124
Sep-74	663,125	673,322
Oct-74	673,323	683,519
Nov-74	683,520	693,717
Dec-74	693,718	703,915
Jan-75	703,916	715,245
Feb-75	715,246	726,575
Mar-75	726,576	737,904
Apr-75	737,905	749,234
May-75	749,235	760,563
Jun-75	760,564	771,893
Jul-75	771,894	783,222
Aug-75	783,223	794,551
Sep-75	794,552	805,881
Oct-75	805,882	817,210
Nov-75	817,211	828,540
Dec-75	828,541	839,869
Jan-76	839,870	852,460
Feb-76	852,461	865,050
Mar-76	865,051	877,640
Apr-76	877,641	890,230
May-76	890,231	902,820
Jun-76	902,821	915,410
Jul-76	915,411	927,999
Aug-76	928,000	940,589

Sep-76	940,590	953,179
Oct-76	953,180	965,769
Nov-76	965,770	978,359
Dec-76	978,360	990,949
Jan-77	990,950	1,002,215
Feb-77	1,002,216	1,013,481
Mar-77	1,013,482	1,024,746
Apr-77	1,024,747	1,036,011
May-77	1,036,012	1,047,276
Jun-77	1,047,277	1,058,542
Jul-77	1,058,543	1,069,807
Aug-77	1,069,808	1,081,072
Sep-77	1,081,073	1,092,337
Oct-77	1,092,338	1,103,603
Nov-77	1,103,604	1,114,868
Dec-77	1,114,869	1,126,133
Jan-78	1,126,134	1,138,645
Feb-78	1,138,646	1,151,156
Mar-78	1,151,157	1,163,668
Apr-78	1,163,669	1,176,179
May-78	1,176,180	1,188,690
Jun-78	1,188,691	1,201,201
Jul-78	1,201,202	1,213,712
Aug-78	1,213,713	1,226,223
Sep-78	1,226,224	1,238,735
Oct-78	1,238,736	1,251,246
Nov-78	1,251,247	1,263,757
Dec-78	1,263,758	1,276,268
Jan-79	1,276,269	1,289,288
Feb-79	1,289,289	1,302,306
Mar-79	1,302,307	1,315,325
Apr-79	1,315,326	1,328,344
May-79	1,328,345	1,341,362
Jun-79	1,341,363	1,354,381
Jul-79	1,354,382	1,367,400
Aug-79	1,367,401	1,380,418
Sep-79	1,380,419	1,393,437
Oct-79	1,393,438	1,406,456
Nov-79	1,406,457	1,419,474
Dec-79	1,419,475	1,432,493
Jan-80	1,432,494	1,446,857
Feb-80	1,446,858	1,461,219
Mar-80	1,461,220	1,475,582
Apr-80	1,475,583	1,489,944
May-80	1,489,945	1,504,307
Jun-80	1,504,308	1,518,670
Jul-80	1,518,671	1,533,032
Aug-80	1,533,033	1,547,395
Sep-80	1,547,396	1,561,757
Oct-80	1,561,758	1,576,120

Nov-80	1,576,121	1,590,482
Dec-80	1,590,483	1,604,845
Jan-81	1,604,846	1,618,814
Feb-81	1,618,815	1,632,782
Mar-81	1,632,783	1,646,751
Apr-81	1,646,752	1,660,719
May-81	1,660,720	1,674,687
Jun-81	1,674,688	1,688,655
Jul-81	1,688,656	1,702,623
Aug-81	1,702,624	1,716,591
Sep-81	1,716,592	1,730,560
Oct-81	1,730,561	1,744,528
Nov-81	1,744,529	1,758,496
Dec-81	1,758,497	1,772,464
Jan-82	1,772,465	1,788,053
Feb-82	1,788,054	1,803,641
Mar-82	1,803,642	1,819,228
Apr-82	1,819,229	1,834,816
May-82	1,834,817	1,850,404
Jun-82	1,850,405	1,865,992
Jul-82	1,865,993	1,881,579
Aug-82	1,881,580	1,897,167
Sep-82	1,897,168	1,912,755
Oct-82	1,912,756	1,928,343
Nov-82	1,928,344	1,943,930
Dec-82	1,943,931	1,959,518
Jan-83	1,959,519	1,975,701
Feb-83	1,975,702	1,991,884
Mar-83	1,991,885	2,008,066
Apr-83	2,008,067	2,024,249
May-83	2,024,250	2,040,431
Jun-83	2,040,432	2,056,614
Jul-83	2,056,615	2,072,796
Aug-83	2,072,797	2,088,978
Sep-83	2,088,979	2,105,161
Oct-83	2,105,162	2,121,343
Nov-83	2,121,344	2,137,526
Dec-83	2,137,527	2,153,708
Jan-84	2,153,709	2,169,988
Feb-84	2,169,989	2,186,268
Mar-84	2,186,269	2,202,547
Apr-84	2,202,548	2,218,827
May-84	2,218,828	2,235,106
Jun-84	2,235,107	2,251,386
Jul-84	2,251,387	2,267,665
Aug-84	2,267,666	2,283,944
Sep-84	2,283,945	2,300,224
Oct-84	2,300,225	2,316,503
Nov-84	2,316,504	2,332,783
Dec-84	2,332,784	2,349,062

Jan-85	2,349,063	2,366,362
Feb-85	2,366,363	2,383,660
Mar-85	2,383,661	2,400,959
Apr-85	2,400,960	2,418,258
May-85	2,418,259	2,435,556
Jun-85	2,435,557	2,452,855
Jul-85	2,452,856	2,470,154
Aug-85	2,470,155	2,487,452
Sep-85	2,487,453	2,504,751
Oct-85	2,504,752	2,522,050
Nov-85	2,522,051	2,539,348
Dec-85	2,539,349	2,556,647
Jan-86	2,556,648	2,574,834
Feb-86	2,574,835	2,593,019
Mar-86	2,593,020	2,611,205
Apr-86	2,611,206	2,629,391
May-86	2,629,392	2,647,576
Jun-86	2,647,577	2,665,762
Jul-86	2,665,763	2,683,948
Aug-86	2,683,949	2,702,133
Sep-86	2,702,134	2,720,319
Oct-86	2,720,320	2,738,505
Nov-86	2,738,506	2,756,690
Dec-86	2,756,691	2,774,876
Jan-87	2,774,877	2,795,403
Feb-87	2,795,404	2,815,929
Mar-87	2,815,930	2,836,455
Apr-87	2,836,456	2,856,981
May-87	2,856,982	2,877,507
Jun-87	2,877,508	2,898,033
Jul-87	2,898,034	2,918,559
Aug-87	2,918,560	2,939,085
Sep-87	2,939,086	2,959,611
Oct-87	2,959,612	2,980,137
Nov-87	2,980,138	3,000,663
Dec-87	3,000,664	3,021,189
Jan-88	3,021,190	3,041,067
Feb-88	3,041,068	3,060,944
Mar-88	3,060,945	3,080,820
Apr-88	3,080,821	3,100,697
May-88	3,100,698	3,120,574
Jun-88	3,120,575	3,140,451
Jul-88	3,140,452	3,160,327
Aug-88	3,160,328	3,180,204
Sep-88	3,180,205	3,200,081
Oct-88	3,200,082	3,219,958
Nov-88	3,219,959	3,239,834
Dec-88	3,239,835	3,259,711
Jan-89	3,259,712	3,280,286
Feb-89	3,280,287	3,300,859

Mar-89	3,300,860	3,321,433
Apr-89	3,321,434	3,342,006
May-89	3,342,007	3,362,580
Jun-89	3,362,581	3,383,153
Jul-89	3,383,154	3,403,727
Aug-89	3,403,728	3,424,300
Sep-89	3,424,301	3,444,874
Oct-89	3,444,875	3,465,447
Nov-89	3,465,448	3,486,021
Dec-89	3,486,022	3,506,594
Jan-90	3,506,595	3,528,124
Feb-90	3,528,125	3,549,653
Mar-90	3,549,654	3,571,182
Apr-90	3,571,183	3,592,711
May-90	3,592,712	3,614,240
Jun-90	3,614,241	3,635,769
Jul-90	3,635,770	3,657,297
Aug-90	3,657,298	3,678,826
Sep-90	3,678,827	3,700,355
Oct-90	3,700,356	3,721,884
Nov-90	3,721,885	3,743,413
Dec-90	3,743,414	3,764,942
Jan-91	3,764,943	3,786,594
Feb-91	3,786,595	3,808,245
Mar-91	3,808,246	3,829,896
Apr-91	3,829,897	3,851,547
May-91	3,851,548	3,873,198
Jun-91	3,873,199	3,894,849
Jul-91	3,894,850	3,916,499
Aug-91	3,916,500	3,938,150
Sep-91	3,938,151	3,959,801
Oct-91	3,959,802	3,981,452
Nov-91	3,981,453	4,003,103
Dec-91	4,003,104	4,024,754
Jan-92	4,024,755	4,049,002
Feb-92	4,049,003	4,073,248
Mar-92	4,073,249	4,097,495
Apr-92	4,097,496	4,121,741
May-92	4,121,742	4,145,988
Jun-92	4,145,989	4,170,235
Jul-92	4,170,236	4,194,481
Aug-92	4,194,482	4,218,728
Sep-92	4,218,729	4,242,974
Oct-92	4,242,975	4,267,221
Nov-92	4,267,222	4,291,467
Dec-92	4,291,468	4,315,714
Jan-93	4,315,715	4,336,048
Feb-93	4,336,049	4,356,381
Mar-93	4,356,382	4,376,714
Apr-93	4,376,715	4,397,047

May-93	4,397,048	4,417,380
Jun-93	4,417,381	4,437,713
Jul-93	4,437,714	4,458,046
Aug-93	4,458,047	4,478,379
Sep-93	4,478,380	4,498,712
Oct-93	4,498,713	4,519,045
Nov-93	4,519,046	4,539,378
Dec-93	4,539,379	4,559,711
Jan-94	4,559,712	4,583,124
Feb-94	4,583,125	4,606,536
Mar-94	4,606,537	4,629,947
Apr-94	4,629,948	4,653,359
May-94	4,653,360	4,676,771
Jun-94	4,676,772	4,700,183
Jul-94	4,700,184	4,723,594
Aug-94	4,723,595	4,747,006
Sep-94	4,747,007	4,770,418
Oct-94	4,770,419	4,793,830
Nov-94	4,793,831	4,817,241
Dec-94	4,817,242	4,840,653
Jan-95	4,840,654	4,866,069
Feb-95	4,866,070	4,891,484
Mar-95	4,891,485	4,916,900
Apr-95	4,916,901	4,942,315
May-95	4,942,316	4,967,730
Jun-95	4,967,731	4,993,145
Jul-95	4,993,146	5,018,560
Aug-95	5,018,561	5,043,975
Sep-95	5,043,976	5,069,391
Oct-95	5,069,392	5,094,806
Nov-95	5,094,807	5,120,221
Dec-95	5,120,222	5,145,636
Jan-96	5,145,637	5,171,599
Feb-96	5,171,600	5,197,561
Mar-96	5,197,562	5,223,522
Apr-96	5,223,523	5,249,484
May-96	5,249,485	5,275,446
Jun-96	5,275,447	5,301,408
Jul-96	5,301,409	5,327,369
Aug-96	5,327,370	5,353,331
Sep-96	5,353,332	5,379,293
Oct-96	5,379,294	5,405,255
Nov-96	5,405,256	5,431,216
Dec-96	5,431,217	5,457,178
Jan-97	5,457,179	5,484,726
Feb-97	5,484,727	5,512,273
Mar-97	5,512,274	5,539,819
Apr-97	5,539,820	5,567,366
May-97	5,567,367	5,594,913
Jun-97	5,594,914	5,622,460

Jul-97	5,622,461	5,650,006
Aug-97	5,650,007	5,677,553
Sep-97	5,677,554	5,705,100
Oct-97	5,705,101	5,732,647
Nov-97	5,732,648	5,760,193
Dec-97	5,760,194	5,787,740
Jan-98	5,787,741	5,816,212
Feb-98	5,816,213	5,844,684
Mar-98	5,844,685	5,873,155
Apr-98	5,873,156	5,901,627
May-98	5,901,628	5,930,098
Jun-98	5,930,099	5,958,570
Jul-98	5,958,571	5,987,041
Aug-98	5,987,042	6,015,512
Sep-98	6,015,513	6,043,984
Oct-98	6,043,985	6,072,455
Nov-98	6,072,456	6,100,927
Dec-98	6,100,928	6,129,398
Jan-99	6,129,399	6,158,933
Feb-99	6,158,934	6,188,467
Mar-99	6,188,468	6,218,001
Apr-99	6,218,002	6,247,535
May-99	6,247,536	6,277,069
Jun-99	6,277,070	6,306,603
Jul-99	6,306,604	6,336,137
Aug-99	6,336,138	6,365,671
Sep-99	6,365,672	6,395,205
Oct-99	6,395,206	6,424,739
Nov-99	6,424,740	6,454,273
Dec-99	6,454,274	6,483,807
Jan-00	6,483,808	6,509,866
Feb-00	6,509,867	6,535,924
Mar-00	6,535,925	6,561,981
Apr-00	6,561,982	6,588,039
May-00	6,588,040	6,614,097
Jun-00	6,614,098	6,640,155
Jul-00	6,640,156	6,666,212
Aug-00	6,666,213	6,692,270
Sep-00	6,692,271	6,718,328
Oct-00	6,718,329	6,744,386
Nov-00	6,744,387	6,770,443
Dec-00	6,770,444	6,796,501

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX M – TERM COMBINATION MACRO CODE

```
'*******************************
'   Macro: sortTrigger
'   Description: This macro uses the descriptors sheet as an input and calls
'   subroutines to combine the n-tuple terms where
'       1-tuple is the sheet of single descriptors
'       2-tuple is the sheet of doubles
'       3-tuple is the sheet of triples
'       etc.
'
'   this macro handles up to 32 tuple
'
'*******************************

Sub sortTrigger()

    descriptorSheet = "n-Terms-01"
    'descriptorSheet = "Descriptors (Cleaned)"
    'descriptorSheet = "Top4Records"
    'descriptorSheet = "Top32Records"

    'Accession number range -- First year --> Last time step
    AN_Start = 0
    AN_End = 116796501

    Call doublesort(descriptorSheet, AN_Start, AN_End)
    Call triplesort(descriptorSheet, AN_Start, AN_End)
    Call quadsort(descriptorSheet, AN_Start, AN_End)
    Call fivesort(descriptorSheet, AN_Start, AN_End)
    Call sixsort(descriptorSheet, AN_Start, AN_End)
    Call sevensort(descriptorSheet, AN_Start, AN_End)
    Call eightsort(descriptorSheet, AN_Start, AN_End)
    Call ninesort(descriptorSheet, AN_Start, AN_End)
    Call tensort(descriptorSheet, AN_Start, AN_End)
    Call elevensort(descriptorSheet, AN_Start, AN_End)
    Call twelvesort(descriptorSheet, AN_Start, AN_End)
    Call thirteensort(descriptorSheet, AN_Start, AN_End)
    Call fourteensort(descriptorSheet, AN_Start, AN_End)
    Call fifteensort(descriptorSheet, AN_Start, AN_End)
    Call sixteensort(descriptorSheet, AN_Start, AN_End)
    Call seventeensort(descriptorSheet, AN_Start, AN_End)
    Call eighteensort(descriptorSheet, AN_Start, AN_End)
    Call Nineteensort(descriptorSheet, AN_Start, AN_End)
    Call twentysort(descriptorSheet, AN_Start, AN_End)
    Call twenty1sort(descriptorSheet, AN_Start, AN_End)
    Call twenty2sort(descriptorSheet, AN_Start, AN_End)
    Call twenty3sort(descriptorSheet, AN_Start, AN_End)
    Call twenty4sort(descriptorSheet, AN_Start, AN_End)
    Call twenty5sort(descriptorSheet, AN_Start, AN_End)
    Call twenty6sort(descriptorSheet, AN_Start, AN_End)
    Call twenty7sort(descriptorSheet, AN_Start, AN_End)
    Call twenty8sort(descriptorSheet, AN_Start, AN_End)
    Call twenty9sort(descriptorSheet, AN_Start, AN_End)
```

```
   Call thirtysort(descriptorSheet, AN_Start, AN_End)
   Call thirty1sort(descriptorSheet, AN_Start, AN_End)
   Call thirty2sort(descriptorSheet, AN_Start, AN_End)

End Sub
Sub sortTriggerMore25()

descriptorSheet = "Descriptors (Cleaned)"

Call twenty6sort(descriptorSheet, AN_Start, AN_End)
Call twenty7sort(descriptorSheet, AN_Start, AN_End)
Call twenty8sort(descriptorSheet, AN_Start, AN_End)
Call twenty9sort(descriptorSheet, AN_Start, AN_End)
Call thirtysort(descriptorSheet, AN_Start, AN_End)
Call thirty1sort(descriptorSheet, AN_Start, AN_End)
Call thirty2sort(descriptorSheet, AN_Start, AN_End)

End Sub

Sub doublesort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As Double)

accession = descriptorSheet
doubles = "n-Terms-02"

Sheets.Add after:=Sheets(Sheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = doubles

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new double sheet



For i = 2 To nRowsAN
   j = i + 1
   accessionNum = Sheets(accession).Cells(i, 1)
'   If (accessionNum >= AN_Start And accessionNum <= AN_End) Then
      Do Until Not (accessionNum = Sheets(accession).Cells(j, 1))
         Sheets(doubles).Cells(k, 1) = accessionNum
         Sheets(doubles).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " & Sheets(accession).Cells(j, 2)
         k = k + 1
         j = j + 1
         If k = 65535 Then
            doubles = "n-Terms-02_2"
            Sheets.Add after:=Sheets(Sheets.Count)
            Sheets(Sheets.Count).Select
            ActiveSheet.Name = doubles
            k = 1
         End If
      Loop
'   End If
Next i

End Sub
Sub triplesort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As Double)
```

```
accession = descriptorSheet
triple = "n-Terms-03"

Sheets.Add after:=Sheets(Sheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = triple

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for triple sheet

For i = 2 To nRowsAN
   j = i + 1
   l = j + 1
   accessionNum = Sheets(accession).Cells(i, 1)

   'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then
      Do Until Not (accessionNum = Sheets(accession).Cells(l, 1))
         Sheets(triple).Cells(k, 1) = accessionNum
         Sheets(triple).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " & Sheets(accession).Cells(j, 2) & " ;
" & Sheets(accession).Cells(l, 2)
         k = k + 1
         j = j + 1
         l = l + 1
         If k = 65500 Then
            triple = "n-Terms-03_2"
            Sheets.Add after:=Sheets(Sheets.Count)
            Sheets(Sheets.Count).Select
            ActiveSheet.Name = triple
            k = 1
         End If

      Loop
   'End If
Next i

End Sub
Sub quadsort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As Double)

accession = descriptorSheet
quad = "n-Terms-04"

Sheets.Add after:=Sheets(Sheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = quad

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new quad combo sheet

For i = 2 To nRowsAN
   j = i + 1
   l = j + 1
   m = l + 1
   accessionNum = Sheets(accession).Cells(i, 1)
```

```
   'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then

       Do Until Not (accessionNum = Sheets(accession).Cells(j, 1) And accessionNum =
Sheets(accession).Cells(l, 1) And accessionNum = Sheets(accession).Cells(m, 1))
          Sheets(quad).Cells(k, 1) = accessionNum
          Sheets(quad).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " & Sheets(accession).Cells(j, 2) & " ;
" & Sheets(accession).Cells(l, 2) & " ; " & Sheets(accession).Cells(m, 2)
          k = k + 1
          j = j + 1
          l = l + 1
          m = m + 1
          If k = 65500 Then
             quad = "n-Terms-04_2"
             Sheets.Add after:=Sheets(Sheets.Count)
             Sheets(Sheets.Count).Select
             ActiveSheet.Name = quad
             k = 1
          End If
       Loop
   'End If
Next i
End Sub

Sub fivesort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As Double)

accession = descriptorSheet
pent = "n-Terms-05"

Sheets.Add after:=Sheets(Sheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = pent

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new quad combo sheet

For i = 2 To nRowsAN
   j = i + 1
   l = j + 1
   m = l + 1
   n = m + 1
   accessionNum = Sheets(accession).Cells(i, 1)
   'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then
       Do Until Not (accessionNum = Sheets(accession).Cells(j, 1) And accessionNum =
Sheets(accession).Cells(l, 1) And accessionNum = Sheets(accession).Cells(m, 1) And accessionNum =
Sheets(accession).Cells(n, 1))
          Sheets(pent).Cells(k, 1) = accessionNum
          Sheets(pent).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " & Sheets(accession).Cells(j, 2) & " ;
" & Sheets(accession).Cells(l, 2) & " ; " & Sheets(accession).Cells(m, 2) & " ; " &
Sheets(accession).Cells(n, 2)
          k = k + 1
          j = j + 1
          l = l + 1
          m = m + 1
          n = n + 1
          If k = 65500 Then
```

```
            pent = "n-Terms-05_2"
            Sheets.Add after:=Sheets(Sheets.Count)
            Sheets(Sheets.Count).Select
            ActiveSheet.Name = pent
            k = 1
          End If

      Loop
   'End If
Next i

End Sub
Sub sixsort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As Double)

accession = descriptorSheet
six = "n-Terms-06"

Sheets.Add after:=Sheets(Sheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = six

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new quad combo sheet

For i = 2 To nRowsAN
   j = i + 1
   l = j + 1
   m = l + 1
   n = m + 1
   o = n + 1
   accessionNum = Sheets(accession).Cells(i, 1)
   'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then
      Do Until Not (accessionNum = Sheets(accession).Cells(j, 1) And accessionNum =
Sheets(accession).Cells(l, 1) And accessionNum = Sheets(accession).Cells(m, 1) And accessionNum =
Sheets(accession).Cells(n, 1) And accessionNum = Sheets(accession).Cells(o, 1))
          Sheets(six).Cells(k, 1) = accessionNum
          Sheets(six).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " & Sheets(accession).Cells(j, 2) & " ; "
& Sheets(accession).Cells(l, 2) & " ; " & Sheets(accession).Cells(m, 2) & " ; " & Sheets(accession).Cells(n,
2) & " ; " & Sheets(accession).Cells(o, 2)
          k = k + 1
          j = j + 1
          l = l + 1
          m = m + 1
          n = n + 1
          o = o + 1
          If k = 65500 Then
            six = "n-Terms-06_2"
            Sheets.Add after:=Sheets(Sheets.Count)
            Sheets(Sheets.Count).Select
            ActiveSheet.Name = six
            k = 1
          End If
      Loop
   'End If
Next i
```

```
End Sub
Sub sevensort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As Double)

accession = descriptorSheet
seven = "n-Terms-07"

Sheets.Add after:=Sheets(Sheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = seven

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new quad combo sheet

For i = 2 To nRowsAN
   j = i + 1
   l = j + 1
   m = l + 1
   n = m + 1
   o = n + 1
   p = o + 1
   accessionNum = Sheets(accession).Cells(i, 1)
   'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then
      Do Until Not (accessionNum = Sheets(accession).Cells(j, 1) And accessionNum =
Sheets(accession).Cells(l, 1) And accessionNum = Sheets(accession).Cells(m, 1) And accessionNum =
Sheets(accession).Cells(n, 1) And accessionNum = Sheets(accession).Cells(o, 1) And accessionNum =
Sheets(accession).Cells(p, 1))
         Sheets(seven).Cells(k, 1) = accessionNum
         Sheets(seven).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " & Sheets(accession).Cells(j, 2) & " ;
" & Sheets(accession).Cells(l, 2) & " ; " & Sheets(accession).Cells(m, 2) & " ; " &
Sheets(accession).Cells(n, 2) & " ; " & Sheets(accession).Cells(o, 2) & " ; " & Sheets(accession).Cells(p, 2)
         k = k + 1
         j = j + 1
         l = l + 1
         m = m + 1
         n = n + 1
         o = o + 1
         p = p + 1
      Loop
   'End If
Next i

End Sub
Sub eightsort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As Double)

accession = descriptorSheet
eight = "n-Terms-08"

Sheets.Add after:=Sheets(Sheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = eight

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new quad combo sheet
```

```
For i = 2 To nRowsAN
   j = i + 1
   l = j + 1
   m = l + 1
   n = m + 1
   o = n + 1
   p = o + 1
   q = p + 1
   accessionNum = Sheets(accession).Cells(i, 1)
   'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then
      Do Until Not (accessionNum = Sheets(accession).Cells(j, 1) And accessionNum =
Sheets(accession).Cells(l, 1) And accessionNum = Sheets(accession).Cells(m, 1) And accessionNum =
Sheets(accession).Cells(n, 1) And accessionNum = Sheets(accession).Cells(o, 1) And accessionNum =
Sheets(accession).Cells(p, 1) And accessionNum = Sheets(accession).Cells(q, 1))
         Sheets(eight).Cells(k, 1) = accessionNum
         Sheets(eight).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " & Sheets(accession).Cells(j, 2) & " ;
" & Sheets(accession).Cells(l, 2) & " ; " & Sheets(accession).Cells(m, 2) & " ; " &
Sheets(accession).Cells(n, 2) & " ; " & Sheets(accession).Cells(o, 2) & " ; " & Sheets(accession).Cells(p, 2)
& " ; " & Sheets(accession).Cells(q, 2)
         k = k + 1
         j = j + 1
         l = l + 1
         m = m + 1
         n = n + 1
         o = o + 1
         p = p + 1
         q = q + 1
      Loop
   'End If
Next i

End Sub
Sub ninesort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As Double)

accession = descriptorSheet
nine = "n-Terms-09"

Sheets.Add after:=Sheets(Sheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = nine

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new quad combo sheet

For i = 2 To nRowsAN
   j = i + 1
   l = j + 1
   m = l + 1
   n = m + 1
   o = n + 1
   p = o + 1
   q = p + 1
   r = q + 1
   accessionNum = Sheets(accession).Cells(i, 1)
```

```
   'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then

       Do Until Not (accessionNum = Sheets(accession).Cells(j, 1) And accessionNum =
Sheets(accession).Cells(l, 1) And accessionNum = Sheets(accession).Cells(m, 1) And accessionNum =
Sheets(accession).Cells(n, 1) And accessionNum = Sheets(accession).Cells(o, 1) And accessionNum =
Sheets(accession).Cells(p, 1) And accessionNum = Sheets(accession).Cells(q, 1) And accessionNum =
Sheets(accession).Cells(r, 1))
          Sheets(nine).Cells(k, 1) = accessionNum
          Sheets(nine).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " & Sheets(accession).Cells(j, 2) & " ;
" & Sheets(accession).Cells(l, 2) & " ; " & Sheets(accession).Cells(m, 2) & " ; " &
Sheets(accession).Cells(n, 2) & " ; " & Sheets(accession).Cells(o, 2) & " ; " & Sheets(accession).Cells(p, 2)
& " ; " & Sheets(accession).Cells(q, 2) & " ; " & Sheets(accession).Cells(r, 2)
          k = k + 1
          j = j + 1
          l = l + 1
          m = m + 1
          n = n + 1
          o = o + 1
          p = p + 1
          q = q + 1
          r = r + 1
       Loop
   'End If
Next i

End Sub
Sub tensort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As Double)

accession = descriptorSheet
ten = "n-Terms-10"

Sheets.Add after:=Sheets(Sheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = ten

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new quad combo sheet

For i = 2 To nRowsAN
   j = i + 1
   l = j + 1
   m = l + 1
   n = m + 1
   o = n + 1
   p = o + 1
   q = p + 1
   r = q + 1
   s = r + 1
   accessionNum = Sheets(accession).Cells(i, 1)

   'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then
       Do Until Not (accessionNum = Sheets(accession).Cells(j, 1) And accessionNum =
Sheets(accession).Cells(l, 1) And accessionNum = Sheets(accession).Cells(m, 1) And accessionNum =
Sheets(accession).Cells(n, 1) And accessionNum = Sheets(accession).Cells(o, 1) And accessionNum =
```

```
Sheets(accession).Cells(p, 1) And accessionNum = Sheets(accession).Cells(q, 1) And accessionNum =
Sheets(accession).Cells(r, 1) And accessionNum = Sheets(accession).Cells(s, 1))
          Sheets(ten).Cells(k, 1) = accessionNum
         Sheets(ten).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " & Sheets(accession).Cells(j, 2) & " ; "
& Sheets(accession).Cells(l, 2) & " ; " & Sheets(accession).Cells(m, 2) & " ; " & Sheets(accession).Cells(n,
2) & " ; " & Sheets(accession).Cells(o, 2) & " ; " & Sheets(accession).Cells(p, 2) & " ; " &
Sheets(accession).Cells(q, 2) & " ; " & Sheets(accession).Cells(r, 2) & " ; " & Sheets(accession).Cells(s, 2)
         k = k + 1
         j = j + 1
         l = l + 1
         m = m + 1
         n = n + 1
         o = o + 1
         p = p + 1
         q = q + 1
         r = r + 1
         s = s + 1
      Loop
   'End If
Next i

End Sub
Sub elevensort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As Double)

accession = descriptorSheet
eleven = "n-Terms-11"

Sheets.Add after:=Sheets(Sheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = eleven

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new quad combo sheet

For i = 2 To nRowsAN
   j = i + 1
   l = j + 1
   m = l + 1
   n = m + 1
   o = n + 1
   p = o + 1
   q = p + 1
   r = q + 1
   s = r + 1
   t = s + 1
   accessionNum = Sheets(accession).Cells(i, 1)

   'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then
      Do Until Not (accessionNum = Sheets(accession).Cells(j, 1) And accessionNum =
Sheets(accession).Cells(l, 1) And accessionNum = Sheets(accession).Cells(m, 1) And accessionNum =
Sheets(accession).Cells(n, 1) And accessionNum = Sheets(accession).Cells(o, 1) And accessionNum =
Sheets(accession).Cells(p, 1) And accessionNum = Sheets(accession).Cells(q, 1) And accessionNum =
Sheets(accession).Cells(r, 1) And accessionNum = Sheets(accession).Cells(s, 1) And accessionNum =
Sheets(accession).Cells(t, 1))
         Sheets(eleven).Cells(k, 1) = accessionNum
```

```
        Sheets(eleven).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " & Sheets(accession).Cells(j, 2) & "
; " & Sheets(accession).Cells(l, 2) & " ; " & Sheets(accession).Cells(m, 2) & " ; " &
Sheets(accession).Cells(n, 2) & " ; " & Sheets(accession).Cells(o, 2) & " ; " & Sheets(accession).Cells(p, 2)
& " ; " & Sheets(accession).Cells(q, 2) & " ; " & Sheets(accession).Cells(r, 2) & " ; " &
Sheets(accession).Cells(s, 2) & " ; " & Sheets(accession).Cells(t, 2)
        k = k + 1
        j = j + 1
        l = l + 1
        m = m + 1
        n = n + 1
        o = o + 1
        p = p + 1
        q = q + 1
        r = r + 1
        s = s + 1
        t = t + 1
      Loop
   'End If
Next i

End Sub
Sub twelvesort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As Double)

accession = descriptorSheet
twelve = "n-Terms-12"

Sheets.Add after:=Sheets(Sheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = twelve

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new quad combo sheet

For i = 2 To nRowsAN
   j = i + 1
   l = j + 1
   m = l + 1
   n = m + 1
   o = n + 1
   p = o + 1
   q = p + 1
   r = q + 1
   s = r + 1
   t = s + 1
   u = t + 1
   accessionNum = Sheets(accession).Cells(i, 1)
   'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then
   Do Until Not (accessionNum = Sheets(accession).Cells(j, 1) And accessionNum =
Sheets(accession).Cells(l, 1) And accessionNum = Sheets(accession).Cells(m, 1) And accessionNum =
Sheets(accession).Cells(n, 1) And accessionNum = Sheets(accession).Cells(o, 1) And accessionNum =
Sheets(accession).Cells(p, 1) And accessionNum = Sheets(accession).Cells(q, 1) And accessionNum =
Sheets(accession).Cells(r, 1) And accessionNum = Sheets(accession).Cells(s, 1) And accessionNum =
Sheets(accession).Cells(t, 1) And accessionNum = Sheets(accession).Cells(u, 1))
       Sheets(twelve).Cells(k, 1) = accessionNum
```

```
        Sheets(twelve).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " & Sheets(accession).Cells(j, 2) & " ;
" & Sheets(accession).Cells(l, 2) & " ; " & Sheets(accession).Cells(m, 2) & " ; " &
Sheets(accession).Cells(n, 2) & " ; " & Sheets(accession).Cells(o, 2) & " ; " & Sheets(accession).Cells(p, 2)
& " ; " & Sheets(accession).Cells(q, 2) & " ; " & Sheets(accession).Cells(r, 2) & " ; " &
Sheets(accession).Cells(s, 2) & " ; " & Sheets(accession).Cells(t, 2) & " ; " & Sheets(accession).Cells(u, 2)
        k = k + 1
        j = j + 1
        l = l + 1
        m = m + 1
        n = n + 1
        o = o + 1
        p = p + 1
        q = q + 1
        r = r + 1
        s = s + 1
        t = t + 1
        u = u + 1
    Loop
    'End If
Next i

End Sub
Sub thirteensort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As
Double)

accession = descriptorSheet
thirteen = "n-Terms-13"

Sheets.Add after:=Sheets(Sheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = thirteen

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new quad combo sheet

For i = 2 To nRowsAN
    j = i + 1
    l = j + 1
    m = l + 1
    n = m + 1
    o = n + 1
    p = o + 1
    q = p + 1
    r = q + 1
    s = r + 1
    t = s + 1
    u = t + 1
    v = u + 1
    accessionNum = Sheets(accession).Cells(i, 1)
    'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then
    Do Until Not (accessionNum = Sheets(accession).Cells(j, 1) And accessionNum =
Sheets(accession).Cells(l, 1) And accessionNum = Sheets(accession).Cells(m, 1) And accessionNum =
Sheets(accession).Cells(n, 1) And accessionNum = Sheets(accession).Cells(o, 1) And accessionNum =
Sheets(accession).Cells(p, 1) And accessionNum = Sheets(accession).Cells(q, 1) And accessionNum =
Sheets(accession).Cells(r, 1) And accessionNum = Sheets(accession).Cells(s, 1) And accessionNum =
```

```
Sheets(accession).Cells(t, 1) And accessionNum = Sheets(accession).Cells(u, 1) And accessionNum =
Sheets(accession).Cells(v, 1))
        Sheets(thirteen).Cells(k, 1) = accessionNum
        Sheets(thirteen).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " & Sheets(accession).Cells(j, 2) & " ;
" & Sheets(accession).Cells(l, 2) & " ; " & Sheets(accession).Cells(m, 2) & " ; " &
Sheets(accession).Cells(n, 2) & " ; " & Sheets(accession).Cells(o, 2) & " ; " & Sheets(accession).Cells(p, 2)
& " ; " & Sheets(accession).Cells(q, 2) & " ; " & Sheets(accession).Cells(r, 2) & " ; " &
Sheets(accession).Cells(s, 2) & " ; " & Sheets(accession).Cells(t, 2) & " ; " & Sheets(accession).Cells(u, 2)
& " ; " & Sheets(accession).Cells(v, 2)
        k = k + 1
        j = j + 1
        l = l + 1
        m = m + 1
        n = n + 1
        o = o + 1
        p = p + 1
        q = q + 1
        r = r + 1
        s = s + 1
        t = t + 1
        u = u + 1
        v = v + 1
    Loop
    'End If
Next i

End Sub
Sub fourteensort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As
Double)

accession = descriptorSheet
fourteen = "n-Terms-14"

Sheets.Add after:=Sheets(Sheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = fourteen

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new quad combo sheet

For i = 2 To nRowsAN
    j = i + 1
    l = j + 1
    m = l + 1
    n = m + 1
    o = n + 1
    p = o + 1
    q = p + 1
    r = q + 1
    s = r + 1
    t = s + 1
    u = t + 1
    v = u + 1
    w = v + 1
    accessionNum = Sheets(accession).Cells(i, 1)
```

```
   'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then
   Do Until Not (accessionNum = Sheets(accession).Cells(j, 1) And accessionNum =
Sheets(accession).Cells(l, 1) And accessionNum = Sheets(accession).Cells(m, 1) And accessionNum =
Sheets(accession).Cells(n, 1) And accessionNum = Sheets(accession).Cells(o, 1) And accessionNum =
Sheets(accession).Cells(p, 1) And accessionNum = Sheets(accession).Cells(q, 1) And accessionNum =
Sheets(accession).Cells(r, 1) And accessionNum = Sheets(accession).Cells(s, 1) And accessionNum =
Sheets(accession).Cells(t, 1) And accessionNum = Sheets(accession).Cells(u, 1) And accessionNum =
Sheets(accession).Cells(v, 1) And accessionNum = Sheets(accession).Cells(w, 1))
       Sheets(fourteen).Cells(k, 1) = accessionNum
       Sheets(fourteen).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " & Sheets(accession).Cells(j, 2) & "
; " & Sheets(accession).Cells(l, 2) & " ; " & Sheets(accession).Cells(m, 2) & " ; " &
Sheets(accession).Cells(n, 2) & " ; " & Sheets(accession).Cells(o, 2) & " ; " & Sheets(accession).Cells(p, 2)
& " ; " & Sheets(accession).Cells(q, 2) & " ; " & Sheets(accession).Cells(r, 2) & " ; " &
Sheets(accession).Cells(s, 2) & " ; " & Sheets(accession).Cells(t, 2) & " ; " & Sheets(accession).Cells(u, 2)
& " ; " & Sheets(accession).Cells(v, 2) & " ; " & Sheets(accession).Cells(w, 2)
       k = k + 1
       j = j + 1
       l = l + 1
       m = m + 1
       n = n + 1
       o = o + 1
       p = p + 1
       q = q + 1
       r = r + 1
       s = s + 1
       t = t + 1
       u = u + 1
       v = v + 1
       w = w + 1
   Loop
   'End If
Next i

End Sub
Sub fifteensort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As Double)

accession = descriptorSheet
fifteen = "n-Terms-15"

Sheets.Add after:=Sheets(Sheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = fifteen

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new quad combo sheet

For i = 2 To nRowsAN
   j = i + 1
   l = j + 1
   m = l + 1
   n = m + 1
   o = n + 1
   p = o + 1
   q = p + 1
   r = q + 1
```

```
    s = r + 1
    t = s + 1
    u = t + 1
    v = u + 1
    w = v + 1
    x = w + 1
    accessionNum = Sheets(accession).Cells(i, 1)
    'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then
    Do Until Not (accessionNum = Sheets(accession).Cells(j, 1) And accessionNum =
Sheets(accession).Cells(l, 1) And accessionNum = Sheets(accession).Cells(m, 1) And accessionNum =
Sheets(accession).Cells(n, 1) And accessionNum = Sheets(accession).Cells(o, 1) And accessionNum =
Sheets(accession).Cells(p, 1) And accessionNum = Sheets(accession).Cells(q, 1) And accessionNum =
Sheets(accession).Cells(r, 1) And accessionNum = Sheets(accession).Cells(s, 1) And accessionNum =
Sheets(accession).Cells(t, 1) And accessionNum = Sheets(accession).Cells(u, 1) And accessionNum =
Sheets(accession).Cells(v, 1) And accessionNum = Sheets(accession).Cells(w, 1) And accessionNum =
Sheets(accession).Cells(x, 1))
        Sheets(fifteen).Cells(k, 1) = accessionNum
        Sheets(fifteen).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " & Sheets(accession).Cells(j, 2) & " ;
" & Sheets(accession).Cells(l, 2) & " ; " & Sheets(accession).Cells(m, 2) & " ; " &
Sheets(accession).Cells(n, 2) & " ; " & Sheets(accession).Cells(o, 2) & " ; " & Sheets(accession).Cells(p, 2)
& " ; " & Sheets(accession).Cells(q, 2) & " ; " & Sheets(accession).Cells(r, 2) & " ; " &
Sheets(accession).Cells(s, 2) & " ; " & Sheets(accession).Cells(t, 2) & " ; " & Sheets(accession).Cells(u, 2)
& " ; " & Sheets(accession).Cells(v, 2) & " ; " & Sheets(accession).Cells(w, 2) & " ; " &
Sheets(accession).Cells(x, 2)
        k = k + 1
        j = j + 1
        l = l + 1
        m = m + 1
        n = n + 1
        o = o + 1
        p = p + 1
        q = q + 1
        r = r + 1
        s = s + 1
        t = t + 1
        u = u + 1
        v = v + 1
        w = w + 1
        x = x + 1
    Loop
    'End If
Next i

End Sub
Sub sixteensort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As Double)

accession = descriptorSheet
sixteen = "n-Terms-16"

Sheets.Add after:=Sheets(Sheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = sixteen

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new quad combo sheet
```

```
For i = 2 To nRowsAN
   j = i + 1
   l = j + 1
   m = l + 1
   n = m + 1
   o = n + 1
   p = o + 1
   q = p + 1
   r = q + 1
   s = r + 1
   t = s + 1
   u = t + 1
   v = u + 1
   w = v + 1
   x = w + 1
   y = x + 1
   accessionNum = Sheets(accession).Cells(i, 1)
   'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then
   Do Until Not (accessionNum = Sheets(accession).Cells(j, 1) And accessionNum =
Sheets(accession).Cells(l, 1) And accessionNum = Sheets(accession).Cells(m, 1) And accessionNum =
Sheets(accession).Cells(n, 1) And accessionNum = Sheets(accession).Cells(o, 1) And accessionNum =
Sheets(accession).Cells(p, 1) And accessionNum = Sheets(accession).Cells(q, 1) And accessionNum =
Sheets(accession).Cells(r, 1) And accessionNum = Sheets(accession).Cells(s, 1) And accessionNum =
Sheets(accession).Cells(t, 1) And accessionNum = Sheets(accession).Cells(u, 1) And accessionNum =
Sheets(accession).Cells(v, 1) And accessionNum = Sheets(accession).Cells(w, 1) And accessionNum =
Sheets(accession).Cells(x, 1) And accessionNum = Sheets(accession).Cells(y, 1))
      Sheets(sixteen).Cells(k, 1) = accessionNum
      Sheets(sixteen).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " & Sheets(accession).Cells(j, 2) & " ;
" & Sheets(accession).Cells(l, 2) & " ; " & Sheets(accession).Cells(m, 2) & " ; " &
Sheets(accession).Cells(n, 2) & " ; " & Sheets(accession).Cells(o, 2) & " ; " & Sheets(accession).Cells(p, 2)
& " ; " & Sheets(accession).Cells(q, 2) & " ; " & Sheets(accession).Cells(r, 2) & " ; " &
Sheets(accession).Cells(s, 2) & " ; " & Sheets(accession).Cells(t, 2) & " ; " & Sheets(accession).Cells(u, 2)
& " ; " & Sheets(accession).Cells(v, 2) & " ; " & Sheets(accession).Cells(w, 2) & " ; " &
Sheets(accession).Cells(x, 2) & " ; " & Sheets(accession).Cells(y, 2)
      k = k + 1
      j = j + 1
      l = l + 1
      m = m + 1
      n = n + 1
      o = o + 1
      p = p + 1
      q = q + 1
      r = r + 1
      s = s + 1
      t = t + 1
      u = u + 1
      v = v + 1
      w = w + 1
      x = x + 1
      y = y + 1
   Loop
   'End If
Next i

End Sub
```

```
Sub seventeensort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As
Double)

accession = descriptorSheet
seventeen = "n-Terms-17"

Sheets.Add after:=Sheets(Sheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = seventeen

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new quad combo sheet

For i = 2 To nRowsAN
   j = i + 1
   l = j + 1
   m = l + 1
   n = m + 1
   o = n + 1
   p = o + 1
   q = p + 1
   r = q + 1
   s = r + 1
   t = s + 1
   u = t + 1
   v = u + 1
   w = v + 1
   x = w + 1
   y = x + 1
   z = y + 1
   accessionNum = Sheets(accession).Cells(i, 1)
   'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then
   Do Until Not (accessionNum = Sheets(accession).Cells(j, 1) And accessionNum =
Sheets(accession).Cells(l, 1) And accessionNum = Sheets(accession).Cells(m, 1) And accessionNum =
Sheets(accession).Cells(n, 1) And accessionNum = Sheets(accession).Cells(o, 1) And accessionNum =
Sheets(accession).Cells(p, 1) And accessionNum = Sheets(accession).Cells(q, 1) And accessionNum =
Sheets(accession).Cells(r, 1) And accessionNum = Sheets(accession).Cells(s, 1) And accessionNum =
Sheets(accession).Cells(t, 1) And accessionNum = Sheets(accession).Cells(u, 1) And accessionNum =
Sheets(accession).Cells(v, 1) And accessionNum = Sheets(accession).Cells(w, 1) And accessionNum =
Sheets(accession).Cells(x, 1) And accessionNum = Sheets(accession).Cells(y, 1) And accessionNum =
Sheets(accession).Cells(z, 1))
      Sheets(seventeen).Cells(k, 1) = accessionNum
      Sheets(seventeen).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " & Sheets(accession).Cells(j, 2) &
" ; " & Sheets(accession).Cells(l, 2) & " ; " & Sheets(accession).Cells(m, 2) & " ; " &
Sheets(accession).Cells(n, 2) & " ; " & Sheets(accession).Cells(o, 2) & " ; " & Sheets(accession).Cells(p, 2)
& " ; " & Sheets(accession).Cells(q, 2) & " ; " & Sheets(accession).Cells(r, 2) & " ; " &
Sheets(accession).Cells(s, 2) & " ; " & Sheets(accession).Cells(t, 2) & " ; " & Sheets(accession).Cells(u, 2)
& " ; " & Sheets(accession).Cells(v, 2) & " ; " & Sheets(accession).Cells(w, 2) & " ; " &
Sheets(accession).Cells(x, 2) & " ; " & Sheets(accession).Cells(y, 2) & " ; " & Sheets(accession).Cells(z, 2)
      k = k + 1
      j = j + 1
      l = l + 1
      m = m + 1
      n = n + 1
      o = o + 1
```

```
        p = p + 1
        q = q + 1
        r = r + 1
        s = s + 1
        t = t + 1
        u = u + 1
        v = v + 1
        w = w + 1
        x = x + 1
        y = y + 1
        z = z + 1
    Loop
    'End If
Next i

End Sub
Sub eighteensort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As
Double)

accession = descriptorSheet
eighteen = "n-Terms-18"

Sheets.Add after:=Sheets(Sheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = eighteen

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new quad combo sheet

For i = 2 To nRowsAN
    j = i + 1
    l = j + 1
    m = l + 1
    n = m + 1
    o = n + 1
    p = o + 1
    q = p + 1
    r = q + 1
    s = r + 1
    t = s + 1
    u = t + 1
    v = u + 1
    w = v + 1
    x = w + 1
    y = x + 1
    z = y + 1
    aa = z + 1
    accessionNum = Sheets(accession).Cells(i, 1)
    'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then
    Do Until Not (accessionNum = Sheets(accession).Cells(j, 1) And accessionNum =
Sheets(accession).Cells(l, 1) And accessionNum = Sheets(accession).Cells(m, 1) And accessionNum =
Sheets(accession).Cells(n, 1) And accessionNum = Sheets(accession).Cells(o, 1) And accessionNum =
Sheets(accession).Cells(p, 1) And accessionNum = Sheets(accession).Cells(q, 1) And accessionNum =
Sheets(accession).Cells(r, 1) And accessionNum = Sheets(accession).Cells(s, 1) And accessionNum =
Sheets(accession).Cells(t, 1) And accessionNum = Sheets(accession).Cells(u, 1) And accessionNum =
```

```
Sheets(accession).Cells(v, 1) And accessionNum = Sheets(accession).Cells(w, 1) And accessionNum =
Sheets(accession).Cells(x, 1) And accessionNum = Sheets(accession).Cells(y, 1) And accessionNum =
Sheets(accession).Cells(z, 1) And accessionNum = Sheets(accession).Cells(aa, 1))
        Sheets(eighteen).Cells(k, 1) = accessionNum
        Sheets(eighteen).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " & Sheets(accession).Cells(j, 2) &
" ; " & Sheets(accession).Cells(l, 2) & " ; " & Sheets(accession).Cells(m, 2) & " ; " &
Sheets(accession).Cells(n, 2) & " ; " & Sheets(accession).Cells(o, 2) & " ; " & Sheets(accession).Cells(p, 2)
& " ; " & Sheets(accession).Cells(q, 2) & " ; " & Sheets(accession).Cells(r, 2) & " ; " &
Sheets(accession).Cells(s, 2) & " ; " & Sheets(accession).Cells(t, 2) & " ; " & Sheets(accession).Cells(u, 2)
& " ; " & Sheets(accession).Cells(v, 2) & " ; " & Sheets(accession).Cells(w, 2) & " ; " &
Sheets(accession).Cells(x, 2) & " ; " & Sheets(accession).Cells(y, 2) & " ; " & Sheets(accession).Cells(z, 2)
& " ; " & Sheets(accession).Cells(aa, 2)
        k = k + 1
        j = j + 1
        l = l + 1
        m = m + 1
        n = n + 1
        o = o + 1
        p = p + 1
        q = q + 1
        r = r + 1
        s = s + 1
        t = t + 1
        u = u + 1
        v = v + 1
        w = w + 1
        x = x + 1
        y = y + 1
        z = z + 1
        aa = aa + 1
    Loop
    'End If
Next i

End Sub
Sub Nineteensort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As
Double)

accession = descriptorSheet
nineteen = "n-Terms-19"

Sheets.Add after:=Sheets(Sheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = nineteen

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new quad combo sheet

For i = 2 To nRowsAN
    j = i + 1
    l = j + 1
    m = l + 1
    n = m + 1
    o = n + 1
    p = o + 1
```

```
    q = p + 1
    r = q + 1
    s = r + 1
    t = s + 1
    u = t + 1
    v = u + 1
    w = v + 1
    x = w + 1
    y = x + 1
    z = y + 1
    aa = z + 1
    bb = aa + 1
    accessionNum = Sheets(accession).Cells(i, 1)
    'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then
    Do Until Not (accessionNum = Sheets(accession).Cells(j, 1) And accessionNum =
Sheets(accession).Cells(l, 1) And accessionNum = Sheets(accession).Cells(m, 1) And accessionNum =
Sheets(accession).Cells(n, 1) And accessionNum = Sheets(accession).Cells(o, 1) And accessionNum =
Sheets(accession).Cells(p, 1) And accessionNum = Sheets(accession).Cells(q, 1) And accessionNum =
Sheets(accession).Cells(r, 1) And accessionNum = Sheets(accession).Cells(s, 1) And accessionNum =
Sheets(accession).Cells(t, 1) And accessionNum = Sheets(accession).Cells(u, 1) And accessionNum =
Sheets(accession).Cells(v, 1) And accessionNum = Sheets(accession).Cells(w, 1) And accessionNum =
Sheets(accession).Cells(x, 1) And accessionNum = Sheets(accession).Cells(y, 1) And accessionNum =
Sheets(accession).Cells(z, 1) And accessionNum = Sheets(accession).Cells(aa, 1) And accessionNum =
Sheets(accession).Cells(bb, 1))
        Sheets(nineteen).Cells(k, 1) = accessionNum
        Sheets(nineteen).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " & Sheets(accession).Cells(j, 2) & "
; " & Sheets(accession).Cells(l, 2) & " ; " & Sheets(accession).Cells(m, 2) & " ; " &
Sheets(accession).Cells(n, 2) & " ; " & Sheets(accession).Cells(o, 2) & " ; " & Sheets(accession).Cells(p, 2)
& " ; " & Sheets(accession).Cells(q, 2) & " ; " & Sheets(accession).Cells(r, 2) & " ; " &
Sheets(accession).Cells(s, 2) & " ; " & Sheets(accession).Cells(t, 2) & " ; " & Sheets(accession).Cells(u, 2)
& " ; " & Sheets(accession).Cells(v, 2) & " ; " & Sheets(accession).Cells(w, 2) & " ; " &
Sheets(accession).Cells(x, 2) & " ; " & Sheets(accession).Cells(y, 2) & " ; " & Sheets(accession).Cells(z, 2)
& " ; " & Sheets(accession).Cells(aa, 2) & " ; " & Sheets(accession).Cells(bb, 2)
        k = k + 1
        j = j + 1
        l = l + 1
        m = m + 1
        n = n + 1
        o = o + 1
        p = p + 1
        q = q + 1
        r = r + 1
        s = s + 1
        t = t + 1
        u = u + 1
        v = v + 1
        w = w + 1
        x = x + 1
        y = y + 1
        z = z + 1
        aa = aa + 1
        bb = bb + 1
    Loop
    'End If
Next i
```

```
End Sub
Sub twentysort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As Double)

accession = descriptorSheet
twenty = "n-Terms-20"

Sheets.Add after:=Sheets(Sheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = twenty

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new quad combo sheet

For i = 2 To nRowsAN
   j = i + 1
   l = j + 1
   m = l + 1
   n = m + 1
   o = n + 1
   p = o + 1
   q = p + 1
   r = q + 1
   s = r + 1
   t = s + 1
   u = t + 1
   v = u + 1
   w = v + 1
   x = w + 1
   y = x + 1
   z = y + 1
   aa = z + 1
   bb = aa + 1
   cc = bb + 1
   accessionNum = Sheets(accession).Cells(i, 1)
   'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then
   Do Until Not (accessionNum = Sheets(accession).Cells(j, 1) And accessionNum =
Sheets(accession).Cells(l, 1) And accessionNum = Sheets(accession).Cells(m, 1) And accessionNum =
Sheets(accession).Cells(n, 1) And accessionNum = Sheets(accession).Cells(o, 1) And accessionNum =
Sheets(accession).Cells(p, 1) And accessionNum = Sheets(accession).Cells(q, 1) And accessionNum =
Sheets(accession).Cells(r, 1) And accessionNum = Sheets(accession).Cells(s, 1) And accessionNum =
Sheets(accession).Cells(t, 1) And accessionNum = Sheets(accession).Cells(u, 1) And accessionNum =
Sheets(accession).Cells(v, 1) And accessionNum = Sheets(accession).Cells(w, 1) And accessionNum =
Sheets(accession).Cells(x, 1) And accessionNum = Sheets(accession).Cells(y, 1) And accessionNum =
Sheets(accession).Cells(z, 1) And accessionNum = Sheets(accession).Cells(aa, 1) And accessionNum =
Sheets(accession).Cells(bb, 1) And accessionNum = Sheets(accession).Cells(cc, 1))
      Sheets(twenty).Cells(k, 1) = accessionNum
      Sheets(twenty).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " & Sheets(accession).Cells(j, 2) & " ;
" & Sheets(accession).Cells(l, 2) & " ; " & Sheets(accession).Cells(m, 2) & " ; " &
Sheets(accession).Cells(n, 2) & " ; " & Sheets(accession).Cells(o, 2) & " ; " & Sheets(accession).Cells(p, 2)
& " ; " & Sheets(accession).Cells(q, 2) & " ; " & Sheets(accession).Cells(r, 2) & " ; " &
Sheets(accession).Cells(s, 2) & " ; " & Sheets(accession).Cells(t, 2) & " ; " & Sheets(accession).Cells(u, 2)
& " ; " & Sheets(accession).Cells(v, 2) & " ; " & Sheets(accession).Cells(w, 2) & " ; " &
Sheets(accession).Cells(x, 2) & " ; " & Sheets(accession).Cells(y, 2) & " ; " & Sheets(accession).Cells(z, 2)
& " ; " & Sheets(accession).Cells(aa, 2) & " ; " & Sheets(accession).Cells(bb, 2) & " ; " &
Sheets(accession).Cells(cc, 2)
```

```
        k = k + 1
        j = j + 1
        l = l + 1
        m = m + 1
        n = n + 1
        o = o + 1
        p = p + 1
        q = q + 1
        r = r + 1
        s = s + 1
        t = t + 1
        u = u + 1
        v = v + 1
        w = w + 1
        x = x + 1
        y = y + 1
        z = z + 1
        aa = aa + 1
        bb = bb + 1
        cc = cc + 1
    Loop
    'End If
Next i

End Sub
Sub twenty1sort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As
Double)

accession = descriptorSheet
twenty1 = "n-Terms-21"

Sheets.Add after:=Sheets(Sheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = twenty1

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new quad combo sheet

For i = 2 To nRowsAN
    j = i + 1
    l = j + 1
    m = l + 1
    n = m + 1
    o = n + 1
    p = o + 1
    q = p + 1
    r = q + 1
    s = r + 1
    t = s + 1
    u = t + 1
    v = u + 1
    w = v + 1
    x = w + 1
    y = x + 1
    z = y + 1
```

```
    aa = z + 1
    bb = aa + 1
    cc = bb + 1
    dd = cc + 1
    accessionNum = Sheets(accession).Cells(i, 1)
    'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then
    Do Until Not (accessionNum = Sheets(accession).Cells(dd, 1))
    'Do Until Not (accessionNum = Sheets(accession).Cells(j, 1) And accessionNum =
Sheets(accession).Cells(l, 1) And accessionNum = Sheets(accession).Cells(m, 1) And accessionNum =
Sheets(accession).Cells(n, 1) And accessionNum = Sheets(accession).Cells(o, 1) And accessionNum =
Sheets(accession).Cells(p, 1) And accessionNum = Sheets(accession).Cells(q, 1) And accessionNum =
Sheets(accession).Cells(r, 1) And accessionNum = Sheets(accession).Cells(s, 1) And accessionNum =
Sheets(accession).Cells(t, 1) And accessionNum = Sheets(accession).Cells(u, 1) And accessionNum =
Sheets(accession).Cells(v, 1) And accessionNum = Sheets(accession).Cells(w, 1) And accessionNum =
Sheets(accession).Cells(x, 1) And accessionNum = Sheets(accession).Cells(y, 1) And accessionNum =
Sheets(accession).Cells(z, 1) And accessionNum = Sheets(accession).Cells(aa, 1) And accessionNum =
Sheets(accession).Cells(bb, 1) And accessionNum = Sheets(accession).Cells(cc, 1) And accessionNum =
Sheets(accession).Cells(dd, 1))
        Sheets(twenty1).Cells(k, 1) = accessionNum
        Sheets(twenty1).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " & Sheets(accession).Cells(j, 2) &
" ; " & Sheets(accession).Cells(l, 2) & " ; " & Sheets(accession).Cells(m, 2) & " ; " &
Sheets(accession).Cells(n, 2) & " ; " & Sheets(accession).Cells(o, 2) & " ; " & Sheets(accession).Cells(p, 2)
& " ; " & Sheets(accession).Cells(q, 2) & " ; " & Sheets(accession).Cells(r, 2) & " ; " &
Sheets(accession).Cells(s, 2) & " ; " & Sheets(accession).Cells(t, 2) & " ; " & Sheets(accession).Cells(u, 2)
& " ; " & Sheets(accession).Cells(v, 2) & " ; " & Sheets(accession).Cells(w, 2) & " ; " &
Sheets(accession).Cells(x, 2) & " ; " & Sheets(accession).Cells(y, 2) & " ; " & Sheets(accession).Cells(z, 2)
& " ; " & Sheets(accession).Cells(aa, 2) & " ; " & Sheets(accession).Cells(bb, 2) & " ; " &
Sheets(accession).Cells(cc, 2) & " ; " & Sheets(accession).Cells(dd, 2)
        k = k + 1
        j = j + 1
        l = l + 1
        m = m + 1
        n = n + 1
        o = o + 1
        p = p + 1
        q = q + 1
        r = r + 1
        s = s + 1
        t = t + 1
        u = u + 1
        v = v + 1
        w = w + 1
        x = x + 1
        y = y + 1
        z = z + 1
        aa = aa + 1
        bb = bb + 1
        cc = cc + 1
        dd = dd + 1
    Loop
    'End If
Next i

End Sub
```

```
Sub twenty2sort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As
Double)

accession = descriptorSheet
twenty2 = "n-Terms-22"

Sheets.Add after:=Sheets(Sheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = twenty2

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new quad combo sheet

For i = 2 To nRowsAN
   j = i + 1
   l = j + 1
   m = l + 1
   n = m + 1
   o = n + 1
   p = o + 1
   q = p + 1
   r = q + 1
   s = r + 1
   t = s + 1
   u = t + 1
   v = u + 1
   w = v + 1
   x = w + 1
   y = x + 1
   z = y + 1
   aa = z + 1
   bb = aa + 1
   cc = bb + 1
   dd = cc + 1
   ee = dd + 1
   accessionNum = Sheets(accession).Cells(i, 1)
   'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then
   Do Until Not (accessionNum = Sheets(accession).Cells(ee, 1))
   'Do Until Not (accessionNum = Sheets(accession).Cells(j, 1) And accessionNum =
Sheets(accession).Cells(l, 1) And accessionNum = Sheets(accession).Cells(m, 1) And accessionNum =
Sheets(accession).Cells(n, 1) And accessionNum = Sheets(accession).Cells(o, 1) And accessionNum =
Sheets(accession).Cells(p, 1) And accessionNum = Sheets(accession).Cells(q, 1) And accessionNum =
Sheets(accession).Cells(r, 1) And accessionNum = Sheets(accession).Cells(s, 1) And accessionNum =
Sheets(accession).Cells(t, 1) And accessionNum = Sheets(accession).Cells(u, 1) And accessionNum =
Sheets(accession).Cells(v, 1) And accessionNum = Sheets(accession).Cells(w, 1) And accessionNum =
Sheets(accession).Cells(x, 1) And accessionNum = Sheets(accession).Cells(y, 1) And accessionNum =
Sheets(accession).Cells(z, 1) And accessionNum = Sheets(accession).Cells(aa, 1) And accessionNum =
Sheets(accession).Cells(bb, 1) And accessionNum = Sheets(accession).Cells(cc, 1) And accessionNum =
Sheets(accession).Cells(dd, 1))
      Sheets(twenty2).Cells(k, 1) = accessionNum
      Sheets(twenty2).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " & Sheets(accession).Cells(j, 2) &
" ; " & Sheets(accession).Cells(l, 2) & " ; " & Sheets(accession).Cells(m, 2) & " ; " &
Sheets(accession).Cells(n, 2) & " ; " & Sheets(accession).Cells(o, 2) & " ; " & Sheets(accession).Cells(p, 2)
& " ; " & Sheets(accession).Cells(q, 2) & " ; " & Sheets(accession).Cells(r, 2) & " ; " &
Sheets(accession).Cells(s, 2) & " ; " & Sheets(accession).Cells(t, 2) & " ; " & Sheets(accession).Cells(u, 2)
```

```
& " ; " & Sheets(accession).Cells(v, 2) & " ; " & Sheets(accession).Cells(w, 2) & " ; " &
Sheets(accession).Cells(x, 2) & " ; " & Sheets(accession).Cells(y, 2) & " ; " & Sheets(accession).Cells(z, 2)
& " ; " & Sheets(accession).Cells(aa, 2) & " ; " & Sheets(accession).Cells(bb, 2) & " ; " &
Sheets(accession).Cells(cc, 2) & " ; " & Sheets(accession).Cells(dd, 2) & " ; " &
Sheets(accession).Cells(ee, 2)
        k = k + 1
        j = j + 1
        l = l + 1
        m = m + 1
        n = n + 1
        o = o + 1
        p = p + 1
        q = q + 1
        r = r + 1
        s = s + 1
        t = t + 1
        u = u + 1
        v = v + 1
        w = w + 1
        x = x + 1
        y = y + 1
        z = z + 1
        aa = aa + 1
        bb = bb + 1
        cc = cc + 1
        dd = dd + 1
        ee = ee + 1
    Loop
    'End If
Next i

End Sub
Sub twenty3sort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As
Double)

accession = descriptorSheet
targetSheetName = "n-Terms-23"

Sheets.Add after:=Worksheets(Worksheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = targetSheetName

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new quad combo sheet

For i = 2 To nRowsAN
    j = i + 1
    l = j + 1
    m = l + 1
    n = m + 1
    o = n + 1
    p = o + 1
    q = p + 1
    r = q + 1
    s = r + 1
```

```
    t = s + 1
    u = t + 1
    v = u + 1
    w = v + 1
    x = w + 1
    y = x + 1
    z = y + 1
    aa = z + 1
    bb = aa + 1
    cc = bb + 1
    dd = cc + 1
    ee = dd + 1 '22
    ff = ee + 1
    accessionNum = Sheets(accession).Cells(i, 1)
    'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then
    Do Until Not (accessionNum = Sheets(accession).Cells(ff, 1))
        Sheets(targetSheetName).Cells(k, 1) = accessionNum
        Sheets(targetSheetName).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " &
Sheets(accession).Cells(j, 2) & " ; " & Sheets(accession).Cells(l, 2) & " ; " & Sheets(accession).Cells(m, 2)
& " ; " & Sheets(accession).Cells(n, 2) & " ; " & Sheets(accession).Cells(o, 2) & " ; " &
Sheets(accession).Cells(p, 2) & " ; " & Sheets(accession).Cells(q, 2) & " ; " & Sheets(accession).Cells(r, 2)
& " ; " & Sheets(accession).Cells(s, 2) & " ; " & Sheets(accession).Cells(t, 2) & " ; " &
Sheets(accession).Cells(u, 2) & " ; " & Sheets(accession).Cells(v, 2) & " ; " & Sheets(accession).Cells(w,
2) & " ; " & Sheets(accession).Cells(x, 2) & " ; " & Sheets(accession).Cells(y, 2) & " ; " &
Sheets(accession).Cells(z, 2) & " ; " & Sheets(accession).Cells(aa, 2) & " ; " & Sheets(accession).Cells(bb,
2) & " ; " & Sheets(accession).Cells(cc, 2) & " ; " & Sheets(accession).Cells(dd, 2) & " ; " &
Sheets(accession).Cells(ee, 2) & " ; " & Sheets(accession).Cells(ff, 2)
        k = k + 1
        j = j + 1
        l = l + 1
        m = m + 1
        n = n + 1
        o = o + 1
        p = p + 1
        q = q + 1
        r = r + 1
        s = s + 1
        t = t + 1
        u = u + 1
        v = v + 1
        w = w + 1
        x = x + 1
        y = y + 1
        z = z + 1
        aa = aa + 1
        bb = bb + 1
        cc = cc + 1
        dd = dd + 1
        ee = ee + 1
        ff = ff + 1
    Loop
    'End If
Next i

End Sub '23
```

```
Sub twenty4sort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As
Double)

accession = descriptorSheet
targetSheetName = "n-Terms-24"

Sheets.Add after:=Worksheets(Worksheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = targetSheetName

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new quad combo sheet

For i = 2 To nRowsAN
   j = i + 1
   l = j + 1
   m = l + 1
   n = m + 1
   o = n + 1
   p = o + 1
   q = p + 1
   r = q + 1
   s = r + 1
   t = s + 1
   u = t + 1
   v = u + 1
   w = v + 1
   x = w + 1
   y = x + 1
   z = y + 1
   aa = z + 1
   bb = aa + 1
   cc = bb + 1
   dd = cc + 1
   ee = dd + 1 '22
   ff = ee + 1
   gg = ff + 1 '24
   accessionNum = Sheets(accession).Cells(i, 1)
   'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then
   Do Until Not (accessionNum = Sheets(accession).Cells(gg, 1))
       Sheets(targetSheetName).Cells(k, 1) = accessionNum
       Sheets(targetSheetName).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " &
Sheets(accession).Cells(j, 2) & " ; " & Sheets(accession).Cells(l, 2) & " ; " & Sheets(accession).Cells(m, 2)
& " ; " & Sheets(accession).Cells(n, 2) & " ; " & Sheets(accession).Cells(o, 2) & " ; " &
Sheets(accession).Cells(p, 2) & " ; " & Sheets(accession).Cells(q, 2) & " ; " & Sheets(accession).Cells(r, 2)
& " ; " & Sheets(accession).Cells(s, 2) & " ; " & Sheets(accession).Cells(t, 2) & " ; " &
Sheets(accession).Cells(u, 2) & " ; " & Sheets(accession).Cells(v, 2) & " ; " & Sheets(accession).Cells(w,
2) & " ; " & Sheets(accession).Cells(x, 2) & " ; " & Sheets(accession).Cells(y, 2) & " ; " &
Sheets(accession).Cells(z, 2) & " ; " & Sheets(accession).Cells(aa, 2) & " ; " & Sheets(accession).Cells(bb,
2) & " ; " & Sheets(accession).Cells(cc, 2) & " ; " & Sheets(accession).Cells(dd, 2) & " ; " &
Sheets(accession).Cells(ee, 2) & " ; " & Sheets(accession).Cells(ff, 2) & " ; " & Sheets(accession).Cells(gg,
2)
       k = k + 1
       j = j + 1
       l = l + 1
```

```
        m = m + 1
        n = n + 1
        o = o + 1
        p = p + 1
        q = q + 1
        r = r + 1
        s = s + 1
        t = t + 1
        u = u + 1
        v = v + 1
        w = w + 1
        x = x + 1
        y = y + 1
        z = z + 1
        aa = aa + 1
        bb = bb + 1
        cc = cc + 1
        dd = dd + 1
        ee = ee + 1
        ff = ff + 1
        gg = gg + 1
    Loop
    'End If
Next i

End Sub '24

Sub twenty5sort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As
Double)

accession = descriptorSheet
targetSheetName = "n-Terms-25"

Sheets.Add after:=Worksheets(Worksheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = targetSheetName

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new quad combo sheet

For i = 2 To nRowsAN
    j = i + 1
    l = j + 1
    m = l + 1
    n = m + 1
    o = n + 1
    p = o + 1
    q = p + 1
    r = q + 1
    s = r + 1
    t = s + 1
    u = t + 1
    v = u + 1
    w = v + 1
    x = w + 1
```

```
    y = x + 1
    z = y + 1
    aa = z + 1
    bb = aa + 1
    cc = bb + 1
    dd = cc + 1
    ee = dd + 1 '22
    ff = ee + 1
    gg = ff + 1 '24
    hh = gg + 1 '25
    accessionNum = Sheets(accession).Cells(i, 1)
    'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then
    Do Until Not (accessionNum = Sheets(accession).Cells(hh, 1))
        Sheets(targetSheetName).Cells(k, 1) = accessionNum
        Sheets(targetSheetName).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " &
Sheets(accession).Cells(j, 2) & " ; " & Sheets(accession).Cells(l, 2) & " ; " & Sheets(accession).Cells(m, 2)
& " ; " & Sheets(accession).Cells(n, 2) & " ; " & Sheets(accession).Cells(o, 2) & " ; " &
Sheets(accession).Cells(p, 2) & " ; " & Sheets(accession).Cells(q, 2) & " ; " & Sheets(accession).Cells(r, 2)
& " ; " & Sheets(accession).Cells(s, 2) & " ; " & Sheets(accession).Cells(t, 2) & " ; " &
Sheets(accession).Cells(u, 2) & " ; " & Sheets(accession).Cells(v, 2) & " ; " & Sheets(accession).Cells(w,
2) & " ; " & Sheets(accession).Cells(x, 2) & " ; " & Sheets(accession).Cells(y, 2) & " ; " &
Sheets(accession).Cells(z, 2) & " ; " & Sheets(accession).Cells(aa, 2) & " ; " & Sheets(accession).Cells(bb,
2) & " ; " & Sheets(accession).Cells(cc, 2) & " ; " & Sheets(accession).Cells(dd, 2) & " ; " &
Sheets(accession).Cells(ee, 2) & " ; " & Sheets(accession).Cells(ff, 2) & " ; " & Sheets(accession).Cells(gg,
2) _
            & " ; " & Sheets(accession).Cells(hh, 2)
        k = k + 1
        j = j + 1
        l = l + 1
        m = m + 1
        n = n + 1
        o = o + 1
        p = p + 1
        q = q + 1
        r = r + 1
        s = s + 1
        t = t + 1
        u = u + 1
        v = v + 1
        w = w + 1
        x = x + 1
        y = y + 1
        z = z + 1
        aa = aa + 1
        bb = bb + 1
        cc = cc + 1
        dd = dd + 1
        ee = ee + 1
        ff = ff + 1
        gg = gg + 1
        hh = hh + 1
    Loop
    'End If
Next i

End Sub ' 25
```

```
Sub twenty6sort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As
Double)

accession = descriptorSheet
targetSheetName = "n-Terms-26"

Sheets.Add after:=Worksheets(Worksheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = targetSheetName

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new quad combo sheet

For i = 2 To nRowsAN
   j = i + 1
   l = j + 1
   m = l + 1
   n = m + 1
   o = n + 1
   p = o + 1
   q = p + 1
   r = q + 1
   s = r + 1
   t = s + 1
   u = t + 1
   v = u + 1
   w = v + 1
   x = w + 1
   y = x + 1
   z = y + 1
   aa = z + 1
   bb = aa + 1
   cc = bb + 1
   dd = cc + 1
   ee = dd + 1 '22
   ff = ee + 1
   gg = ff + 1 '24
   hh = gg + 1 '25
   ii = hh + 1 '26
   accessionNum = Sheets(accession).Cells(i, 1)
   'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then
   Do Until Not (accessionNum = Sheets(accession).Cells(ii, 1))
       Sheets(targetSheetName).Cells(k, 1) = accessionNum
       Sheets(targetSheetName).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " &
Sheets(accession).Cells(j, 2) & " ; " & Sheets(accession).Cells(l, 2) & " ; " & Sheets(accession).Cells(m, 2)
& " ; " & Sheets(accession).Cells(n, 2) & " ; " & Sheets(accession).Cells(o, 2) & " ; " &
Sheets(accession).Cells(p, 2) & " ; " & Sheets(accession).Cells(q, 2) & " ; " & Sheets(accession).Cells(r, 2)
& " ; " & Sheets(accession).Cells(s, 2) & " ; " & Sheets(accession).Cells(t, 2) & " ; " &
Sheets(accession).Cells(u, 2) & " ; " & Sheets(accession).Cells(v, 2) & " ; " & Sheets(accession).Cells(w,
2) & " ; " & Sheets(accession).Cells(x, 2) & " ; " & Sheets(accession).Cells(y, 2) & " ; " &
Sheets(accession).Cells(z, 2) & " ; " & Sheets(accession).Cells(aa, 2) & " ; " & Sheets(accession).Cells(bb,
2) & " ; " & Sheets(accession).Cells(cc, 2) & " ; " & Sheets(accession).Cells(dd, 2) & " ; " &
Sheets(accession).Cells(ee, 2) & " ; " & Sheets(accession).Cells(ff, 2) & " ; " & Sheets(accession).Cells(gg,
2) _
         & " ; " & Sheets(accession).Cells(hh, 2) & " ; " & Sheets(accession).Cells(ii, 2)
```

```
        k = k + 1
        j = j + 1
        l = l + 1
        m = m + 1
        n = n + 1
        o = o + 1
        p = p + 1
        q = q + 1
        r = r + 1
        s = s + 1
        t = t + 1
        u = u + 1
        v = v + 1
        w = w + 1
        x = x + 1
        y = y + 1
        z = z + 1
        aa = aa + 1
        bb = bb + 1
        cc = cc + 1
        dd = dd + 1
        ee = ee + 1
        ff = ff + 1
        gg = gg + 1
        hh = hh + 1
        ii = ii + 1
    Loop
    'End If
Next i

End Sub ' 26
Sub twenty7sort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As
Double)

accession = descriptorSheet
targetSheetName = "n-Terms-27"

Sheets.Add after:=Worksheets(Worksheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = targetSheetName

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new quad combo sheet

For i = 2 To nRowsAN
    j = i + 1
    l = j + 1
    m = l + 1
    n = m + 1
    o = n + 1
    p = o + 1
    q = p + 1
    r = q + 1
    s = r + 1
    t = s + 1
```

```
    u = t + 1
    v = u + 1
    w = v + 1
    x = w + 1
    y = x + 1
    z = y + 1
    aa = z + 1
    bb = aa + 1
    cc = bb + 1
    dd = cc + 1
    ee = dd + 1 '22
    ff = ee + 1
    gg = ff + 1 '24
    hh = gg + 1 '25
    ii = hh + 1 '26
    jj = ii + 1 '27
    accessionNum = Sheets(accession).Cells(i, 1)
    'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then
    Do Until Not (accessionNum = Sheets(accession).Cells(jj, 1))
        Sheets(targetSheetName).Cells(k, 1) = accessionNum
        Sheets(targetSheetName).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " &
Sheets(accession).Cells(j, 2) & " ; " & Sheets(accession).Cells(l, 2) & " ; " & Sheets(accession).Cells(m, 2)
& " ; " & Sheets(accession).Cells(n, 2) & " ; " & Sheets(accession).Cells(o, 2) & " ; " &
Sheets(accession).Cells(p, 2) & " ; " & Sheets(accession).Cells(q, 2) & " ; " & Sheets(accession).Cells(r, 2)
& " ; " & Sheets(accession).Cells(s, 2) & " ; " & Sheets(accession).Cells(t, 2) & " ; " &
Sheets(accession).Cells(u, 2) & " ; " & Sheets(accession).Cells(v, 2) & " ; " & Sheets(accession).Cells(w,
2) & " ; " & Sheets(accession).Cells(x, 2) & " ; " & Sheets(accession).Cells(y, 2) & " ; " &
Sheets(accession).Cells(z, 2) & " ; " & Sheets(accession).Cells(aa, 2) & " ; " & Sheets(accession).Cells(bb,
2) & " ; " & Sheets(accession).Cells(cc, 2) & " ; " & Sheets(accession).Cells(dd, 2) & " ; " &
Sheets(accession).Cells(ee, 2) & " ; " & Sheets(accession).Cells(ff, 2) & " ; " & Sheets(accession).Cells(gg,
2) _
          & " ; " & Sheets(accession).Cells(hh, 2) & " ; " & Sheets(accession).Cells(ii, 2) & " ; " &
Sheets(accession).Cells(jj, 2)
        k = k + 1
        j = j + 1
        l = l + 1
        m = m + 1
        n = n + 1
        o = o + 1
        p = p + 1
        q = q + 1
        r = r + 1
        s = s + 1
        t = t + 1
        u = u + 1
        v = v + 1
        w = w + 1
        x = x + 1
        y = y + 1
        z = z + 1
        aa = aa + 1
        bb = bb + 1
        cc = cc + 1
        dd = dd + 1
        ee = ee + 1
        ff = ff + 1
```

```
        gg = gg + 1
        hh = hh + 1
        ii = ii + 1 '26
        jj = jj + 1 '27
    Loop
    'End If
Next i

End Sub ' 27
Sub twenty8sort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As
Double)

accession = descriptorSheet
targetSheetName = "n-Terms-28"

Sheets.Add after:=Worksheets(Worksheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = targetSheetName

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new quad combo sheet

For i = 2 To nRowsAN
    j = i + 1
    l = j + 1
    m = l + 1
    n = m + 1
    o = n + 1
    p = o + 1
    q = p + 1
    r = q + 1
    s = r + 1
    t = s + 1
    u = t + 1
    v = u + 1
    w = v + 1
    x = w + 1
    y = x + 1
    z = y + 1
    aa = z + 1
    bb = aa + 1
    cc = bb + 1
    dd = cc + 1
    ee = dd + 1 '22
    ff = ee + 1
    gg = ff + 1 '24
    hh = gg + 1 '25
    ii = hh + 1 '26
    jj = ii + 1 '27
    kk = jj + 1 '28
    accessionNum = Sheets(accession).Cells(i, 1)
    'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then
    Do Until Not (accessionNum = Sheets(accession).Cells(kk, 1))
        Sheets(targetSheetName).Cells(k, 1) = accessionNum
```

```
        Sheets(targetSheetName).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " &
Sheets(accession).Cells(j, 2) & " ; " & Sheets(accession).Cells(l, 2) & " ; " & Sheets(accession).Cells(m, 2)
& " ; " & Sheets(accession).Cells(n, 2) & " ; " & Sheets(accession).Cells(o, 2) & " ; " &
Sheets(accession).Cells(p, 2) & " ; " & Sheets(accession).Cells(q, 2) & " ; " & Sheets(accession).Cells(r, 2)
& " ; " & Sheets(accession).Cells(s, 2) & " ; " & Sheets(accession).Cells(t, 2) & " ; " &
Sheets(accession).Cells(u, 2) & " ; " & Sheets(accession).Cells(v, 2) & " ; " & Sheets(accession).Cells(w,
2) & " ; " & Sheets(accession).Cells(x, 2) & " ; " & Sheets(accession).Cells(y, 2) & " ; " &
Sheets(accession).Cells(z, 2) & " ; " & Sheets(accession).Cells(aa, 2) & " ; " & Sheets(accession).Cells(bb,
2) & " ; " & Sheets(accession).Cells(cc, 2) & " ; " & Sheets(accession).Cells(dd, 2) & " ; " &
Sheets(accession).Cells(ee, 2) & " ; " & Sheets(accession).Cells(ff, 2) & " ; " & Sheets(accession).Cells(gg,
2) _
          & " ; " & Sheets(accession).Cells(hh, 2) & " ; " & Sheets(accession).Cells(ii, 2) & " ; " &
Sheets(accession).Cells(jj, 2) & " ; " & Sheets(accession).Cells(kk, 2)
        k = k + 1
        j = j + 1
        l = l + 1
        m = m + 1
        n = n + 1
        o = o + 1
        p = p + 1
        q = q + 1
        r = r + 1
        s = s + 1
        t = t + 1
        u = u + 1
        v = v + 1
        w = w + 1
        x = x + 1
        y = y + 1
        z = z + 1
        aa = aa + 1
        bb = bb + 1
        cc = cc + 1
        dd = dd + 1
        ee = ee + 1
        ff = ff + 1
        gg = gg + 1
        hh = hh + 1
        ii = ii + 1 '26
        jj = jj + 1 '27
        kk = kk + 1 '28
    Loop
    'End If
Next i

End Sub ' 28
Sub twenty9sort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As
Double)

accession = descriptorSheet
targetSheetName = "n-Terms-29"

Sheets.Add after:=Worksheets(Worksheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = targetSheetName
```

```
nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new quad combo sheet

For i = 2 To nRowsAN
   j = i + 1
   l = j + 1
   m = l + 1
   n = m + 1
   o = n + 1
   p = o + 1
   q = p + 1
   r = q + 1
   s = r + 1
   t = s + 1
   u = t + 1
   v = u + 1
   w = v + 1
   x = w + 1
   y = x + 1
   z = y + 1
   aa = z + 1
   bb = aa + 1
   cc = bb + 1
   dd = cc + 1
   ee = dd + 1 '22
   ff = ee + 1
   gg = ff + 1 '24
   hh = gg + 1 '25
   ii = hh + 1 '26
   jj = ii + 1 '27
   kk = jj + 1 '28
   ll = kk + 1 '29
   accessionNum = Sheets(accession).Cells(i, 1)
   'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then
   Do Until Not (accessionNum = Sheets(accession).Cells(ll, 1))
      Sheets(targetSheetName).Cells(k, 1) = accessionNum
      Sheets(targetSheetName).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " &
Sheets(accession).Cells(j, 2) & " ; " & Sheets(accession).Cells(l, 2) & " ; " & Sheets(accession).Cells(m, 2)
& " ; " & Sheets(accession).Cells(n, 2) & " ; " & Sheets(accession).Cells(o, 2) & " ; " &
Sheets(accession).Cells(p, 2) & " ; " & Sheets(accession).Cells(q, 2) & " ; " & Sheets(accession).Cells(r, 2)
& " ; " & Sheets(accession).Cells(s, 2) & " ; " & Sheets(accession).Cells(t, 2) & " ; " &
Sheets(accession).Cells(u, 2) & " ; " & Sheets(accession).Cells(v, 2) & " ; " & Sheets(accession).Cells(w,
2) & " ; " & Sheets(accession).Cells(x, 2) & " ; " & Sheets(accession).Cells(y, 2) & " ; " &
Sheets(accession).Cells(z, 2) & " ; " & Sheets(accession).Cells(aa, 2) & " ; " & Sheets(accession).Cells(bb,
2) & " ; " & Sheets(accession).Cells(cc, 2) & " ; " & Sheets(accession).Cells(dd, 2) & " ; " &
Sheets(accession).Cells(ee, 2) & " ; " & Sheets(accession).Cells(ff, 2) & " ; " & Sheets(accession).Cells(gg,
2) _
         & " ; " & Sheets(accession).Cells(hh, 2) & " ; " & Sheets(accession).Cells(ii, 2) & " ; " &
Sheets(accession).Cells(jj, 2) & " ; " & Sheets(accession).Cells(kk, 2) & " ; " & Sheets(accession).Cells(ll,
2)
      k = k + 1
      j = j + 1
      l = l + 1
      m = m + 1
      n = n + 1
```

```
        o = o + 1
        p = p + 1
        q = q + 1
        r = r + 1
        s = s + 1
        t = t + 1
        u = u + 1
        v = v + 1
        w = w + 1
        x = x + 1
        y = y + 1
        z = z + 1
        aa = aa + 1
        bb = bb + 1
        cc = cc + 1
        dd = dd + 1
        ee = ee + 1
        ff = ff + 1
        gg = gg + 1
        hh = hh + 1
        ii = ii + 1 '26
        jj = jj + 1 '27
        kk = kk + 1 '28
        ll = ll + 1 '29
    Loop
    'End If
Next i

End Sub ' 29
Sub thirtysort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As Double)

accession = descriptorSheet
targetSheetName = "n-Terms-30"

Sheets.Add after:=Worksheets(Worksheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = targetSheetName

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new quad combo sheet

For i = 2 To nRowsAN
    j = i + 1
    l = j + 1
    m = l + 1
    n = m + 1
    o = n + 1
    p = o + 1
    q = p + 1
    r = q + 1
    s = r + 1
    t = s + 1
    u = t + 1
    v = u + 1
    w = v + 1
```

```
    x = w + 1
    y = x + 1
    z = y + 1
    aa = z + 1
    bb = aa + 1
    cc = bb + 1
    dd = cc + 1
    ee = dd + 1 '22
    ff = ee + 1
    gg = ff + 1 '24
    hh = gg + 1 '25
    ii = hh + 1 '26
    jj = ii + 1 '27
    kk = jj + 1 '28
    ll = kk + 1 '29
    mm = ll + 1 '30
    accessionNum = Sheets(accession).Cells(i, 1)
    'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then
    Do Until Not (accessionNum = Sheets(accession).Cells(mm, 1))
        Sheets(targetSheetName).Cells(k, 1) = accessionNum
        Sheets(targetSheetName).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " &
Sheets(accession).Cells(j, 2) & " ; " & Sheets(accession).Cells(l, 2) & " ; " & Sheets(accession).Cells(m, 2)
& " ; " & Sheets(accession).Cells(n, 2) & " ; " & Sheets(accession).Cells(o, 2) & " ; " &
Sheets(accession).Cells(p, 2) & " ; " & Sheets(accession).Cells(q, 2) & " ; " & Sheets(accession).Cells(r, 2)
& " ; " & Sheets(accession).Cells(s, 2) & " ; " & Sheets(accession).Cells(t, 2) & " ; " &
Sheets(accession).Cells(u, 2) & " ; " & Sheets(accession).Cells(v, 2) & " ; " & Sheets(accession).Cells(w,
2) & " ; " & Sheets(accession).Cells(x, 2) & " ; " & Sheets(accession).Cells(y, 2) & " ; " &
Sheets(accession).Cells(z, 2) & " ; " & Sheets(accession).Cells(aa, 2) & " ; " & Sheets(accession).Cells(bb,
2) & " ; " & Sheets(accession).Cells(cc, 2) & " ; " & Sheets(accession).Cells(dd, 2) & " ; " &
Sheets(accession).Cells(ee, 2) & " ; " & Sheets(accession).Cells(ff, 2) & " ; " & Sheets(accession).Cells(gg,
2) _
            & " ; " & Sheets(accession).Cells(hh, 2) & " ; " & Sheets(accession).Cells(ii, 2) & " ; " &
Sheets(accession).Cells(jj, 2) & " ; " & Sheets(accession).Cells(kk, 2) & " ; " & Sheets(accession).Cells(ll,
2) & " ; " & Sheets(accession).Cells(mm, 2)
        k = k + 1
        j = j + 1
        l = l + 1
        m = m + 1
        n = n + 1
        o = o + 1
        p = p + 1
        q = q + 1
        r = r + 1
        s = s + 1
        t = t + 1
        u = u + 1
        v = v + 1
        w = w + 1
        x = x + 1
        y = y + 1
        z = z + 1
        aa = aa + 1
        bb = bb + 1
        cc = cc + 1
        dd = dd + 1
        ee = ee + 1
```

```
        ff = ff + 1
        gg = gg + 1
        hh = hh + 1
        ii = ii + 1 '26
        jj = jj + 1 '27
        kk = kk + 1 '28
        ll = ll + 1 '29
        mm = mm + 1 '30
    Loop
    'End If
Next i
End Sub ' 30
Sub thirty1sort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As Double)

accession = descriptorSheet
targetSheetName = "n-Terms-31"

Sheets.Add after:=Worksheets(Worksheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = targetSheetName

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new quad combo sheet

For i = 2 To nRowsAN
    j = i + 1
    l = j + 1
    m = l + 1
    n = m + 1
    o = n + 1
    p = o + 1
    q = p + 1
    r = q + 1
    s = r + 1
    t = s + 1
    u = t + 1
    v = u + 1
    w = v + 1
    x = w + 1
    y = x + 1
    z = y + 1
    aa = z + 1
    bb = aa + 1
    cc = bb + 1
    dd = cc + 1
    ee = dd + 1 '22
    ff = ee + 1
    gg = ff + 1 '24
    hh = gg + 1 '25
    ii = hh + 1 '26
    jj = ii + 1 '27
    kk = jj + 1 '28
    ll = kk + 1 '29
    mm = ll + 1 '30
    nn = mm + 1 '31
```

```
    accessionNum = Sheets(accession).Cells(i, 1)
    'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then
    Do Until Not (accessionNum = Sheets(accession).Cells(nn, 1))
        Sheets(targetSheetName).Cells(k, 1) = accessionNum
        Sheets(targetSheetName).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " &
Sheets(accession).Cells(j, 2) & " ; " & Sheets(accession).Cells(l, 2) & " ; " & Sheets(accession).Cells(m, 2)
& " ; " & Sheets(accession).Cells(n, 2) & " ; " & Sheets(accession).Cells(o, 2) & " ; " &
Sheets(accession).Cells(p, 2) & " ; " & Sheets(accession).Cells(q, 2) & " ; " & Sheets(accession).Cells(r, 2)
& " ; " & Sheets(accession).Cells(s, 2) & " ; " & Sheets(accession).Cells(t, 2) & " ; " &
Sheets(accession).Cells(u, 2) & " ; " & Sheets(accession).Cells(v, 2) & " ; " & Sheets(accession).Cells(w,
2) & " ; " & Sheets(accession).Cells(x, 2) & " ; " & Sheets(accession).Cells(y, 2) & " ; " &
Sheets(accession).Cells(z, 2) & " ; " & Sheets(accession).Cells(aa, 2) & " ; " & Sheets(accession).Cells(bb,
2) & " ; " & Sheets(accession).Cells(cc, 2) & " ; " & Sheets(accession).Cells(dd, 2) & " ; " &
Sheets(accession).Cells(ee, 2) & " ; " & Sheets(accession).Cells(ff, 2) & " ; " & Sheets(accession).Cells(gg,
2) _
          & " ; " & Sheets(accession).Cells(hh, 2) & " ; " & Sheets(accession).Cells(ii, 2) & " ; " &
Sheets(accession).Cells(jj, 2) & " ; " & Sheets(accession).Cells(kk, 2) & " ; " & Sheets(accession).Cells(ll,
2) & " ; " & Sheets(accession).Cells(mm, 2) & " ; " & Sheets(accession).Cells(nn, 2)
        k = k + 1
        j = j + 1
        l = l + 1
        m = m + 1
        n = n + 1
        o = o + 1
        p = p + 1
        q = q + 1
        r = r + 1
        s = s + 1
        t = t + 1
        u = u + 1
        v = v + 1
        w = w + 1
        x = x + 1
        y = y + 1
        z = z + 1
        aa = aa + 1
        bb = bb + 1
        cc = cc + 1
        dd = dd + 1
        ee = ee + 1
        ff = ff + 1
        gg = gg + 1
        hh = hh + 1
        ii = ii + 1 '26
        jj = jj + 1 '27
        kk = kk + 1 '28
        ll = ll + 1 '29
        mm = mm + 1 '30
        nn = nn + 1 '31
    Loop
    'End If
Next i

End Sub ' 31
Sub thirty2sort(ByVal descriptorSheet As String, ByVal AN_Start As Double, ByVal AN_End As Double)
```

```
accession = descriptorSheet
targetSheetName = "n-Terms-32"

Sheets.Add after:=Worksheets(Worksheets.Count)
Sheets(Sheets.Count).Select
ActiveSheet.Name = targetSheetName

nRowsAN = CountRows(accession, 1) 'Find number of rows

k = 1 'row counter for new quad combo sheet

For i = 2 To nRowsAN
   j = i + 1
   l = j + 1
   m = l + 1
   n = m + 1
   o = n + 1
   p = o + 1
   q = p + 1
   r = q + 1
   s = r + 1
   t = s + 1
   u = t + 1
   v = u + 1
   w = v + 1
   x = w + 1
   y = x + 1
   z = y + 1
   aa = z + 1
   bb = aa + 1
   cc = bb + 1
   dd = cc + 1
   ee = dd + 1 '22
   ff = ee + 1
   gg = ff + 1 '24
   hh = gg + 1 '25
   ii = hh + 1 '26
   jj = ii + 1 '27
   kk = jj + 1 '28
   ll = kk + 1 '29
   mm = ll + 1 '30
   nn = mm + 1 '31
   oo = nn + 1 '32
   accessionNum = Sheets(accession).Cells(i, 1)
   'If (accessionNum >= AN_Start And accessionNum <= AN_End) Then
   Do Until Not (accessionNum = Sheets(accession).Cells(oo, 1))
      Sheets(targetSheetName).Cells(k, 1) = accessionNum
      Sheets(targetSheetName).Cells(k, 2) = Sheets(accession).Cells(i, 2) & " ; " &
Sheets(accession).Cells(j, 2) & " ; " & Sheets(accession).Cells(l, 2) & " ; " & Sheets(accession).Cells(m, 2)
& " ; " & Sheets(accession).Cells(n, 2) & " ; " & Sheets(accession).Cells(o, 2) & " ; " &
Sheets(accession).Cells(p, 2) & " ; " & Sheets(accession).Cells(q, 2) & " ; " & Sheets(accession).Cells(r, 2)
& " ; " & Sheets(accession).Cells(s, 2) & " ; " & Sheets(accession).Cells(t, 2) & " ; " &
Sheets(accession).Cells(u, 2) & " ; " & Sheets(accession).Cells(v, 2) & " ; " & Sheets(accession).Cells(w,
2) & " ; " & Sheets(accession).Cells(x, 2) & " ; " & Sheets(accession).Cells(y, 2) & " ; " &
Sheets(accession).Cells(z, 2) & " ; " & Sheets(accession).Cells(aa, 2) & " ; " & Sheets(accession).Cells(bb,
2) & " ; " & Sheets(accession).Cells(cc, 2) & " ; " & Sheets(accession).Cells(dd, 2) & " ; " &
```

```
Sheets(accession).Cells(ee, 2) & " ; " & Sheets(accession).Cells(ff, 2) & " ; " & Sheets(accession).Cells(gg, 2) _
        & " ; " & Sheets(accession).Cells(hh, 2) & " ; " & Sheets(accession).Cells(ii, 2) & " ; " & Sheets(accession).Cells(jj, 2) & " ; " & Sheets(accession).Cells(kk, 2) & " ; " & Sheets(accession).Cells(ll, 2) & " ; " & Sheets(accession).Cells(mm, 2) & " ; " & Sheets(accession).Cells(nn, 2) & " ; " & Sheets(accession).Cells(oo, 2)
      k = k + 1
      j = j + 1
      l = l + 1
      m = m + 1
      n = n + 1
      o = o + 1
      p = p + 1
      q = q + 1
      r = r + 1
      s = s + 1
      t = t + 1
      u = u + 1
      v = v + 1
      w = w + 1
      x = x + 1
      y = y + 1
      z = z + 1
      aa = aa + 1
      bb = bb + 1
      cc = cc + 1
      dd = dd + 1
      ee = ee + 1
      ff = ff + 1
      gg = gg + 1
      hh = hh + 1
      ii = ii + 1 '26
      jj = jj + 1 '27
      kk = kk + 1 '28
      ll = ll + 1 '29
      mm = mm + 1 '30
      nn = nn + 1 '31
      oo = oo + 1 '32
   Loop
   'End If
Next i
End Sub ' 32

'*********************************
'  col - returns column letter
'
'   input column number
'
'*********************************

Function col(ByVal columnNumber As Integer) As String

Select Case columnNumber
   Case 1
      col = "A"
   Case 2
```

```
    col = "B"
Case 3
    col = "C"
Case 4
    col = "D"
Case 5
    col = "E"
Case 6
    col = "F"
Case 7
    col = "G"
Case 8
    col = "H"
Case 9
    col = "i"
Case 10
    col = "J"
Case 11
    col = "K"
Case 12
    col = "L"
Case 13
    col = "M"
Case 14
    col = "N"
Case 15
    col = "O"
Case 16
    col = "P"
Case 17
    col = "Q"
Case 18
    col = "R"
Case 19
    col = "S"
Case 20
    col = "T"
Case 21
    col = "U"
Case 22
    col = "V"
Case 23
    col = "W"
Case 24
    col = "X"
Case 25
    col = "Y"
Case 26
    col = "Z"
Case 27
    col = "AA"
Case 28
    col = "AB"
Case 29
    col = "AC"
Case 30
```

```
    col = "AD"
Case 31
    col = "AE"
Case 32
    col = "AF"
Case 33
    col = "AG"
Case 34
    col = "AH"
Case 35
    col = "AI"
Case 36
    col = "AJ"
Case 37
    col = "AK"
Case 38
    col = "AL"
Case 39
    col = "AM"
Case 40
    col = "AN"
Case 41
    col = "AO"
Case 42
    col = "AP"
Case 43
    col = "AQ"
Case 44
    col = "AR"
Case 45
    col = "AS"
Case 46
    col = "AT"
Case 47
    col = "AU"
Case 48
    col = "AV"
Case 49
    col = "AW"
Case 50
    col = "AX"
Case 51
    col = "AY"
Case 52
    col = "AZ"
Case 53
    col = "BA"
Case 54
    col = "BB"
Case 55
    col = "BC"
Case 56
    col = "BD"
Case 57
    col = "BE"
Case others
```

```
        col = "Z"
End Select

End Function

Function CountRows(ByVal sheetName As String, ByVal colNum As Integer) As Double
On Error Resume Next
Dim currCell As Range, rowNum As Double

Sheets("" & sheetName).Select

If IsNumeric(colNum) Then
Else
   colNum = 1
End If
rowNum = 1
Set currCell = ActiveSheet.Cells(rowNum, colNum)
Do While currCell.Value <> ""
   rowNum = rowNum + 1
   Set currCell = ActiveSheet.Cells(rowNum, colNum)
Loop
CountRows = rowNum - 1
End Function
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX N – AFFILIATION DISTRIBUTION MARCO CODE

```
'-------------------------------------------------------------
'  MACRO: AffiliationDistribution
'  Author: Matt Behnke
'  Created: 11/5/01
'  Description: 1) distributes affiliations into bands corresponding to the mean and
'               standard deviation of the number of publications an affiliation produced
'            2)computes the entropy of the bands and the world
'            3)calculations temperature and pressure of the dataset
'
'-------------------------------------------------------------------------

'GLOBAL VARIABLES

Dim technologyName As String                    'name of the dataset (ada, java, etc)
Dim stepInterval As String                      'the time between time steps (months, years)
Dim currFilename As String                      'the name of the spreadsheet file
Dim dataSheet As String                         'sheet that contains the matrix of affiliations
Dim descriptorMatrixSheet As String             'sheet that contains the matrix of terms (X)
Dim descriptorMatrixSheetY As String            'sheet that contains the matrix of terms (opposite of X)
Dim worldEntropySheet As String                 'sheet that contains world entropy
Dim worldEntropySheetY As String                'sheet that contains world entropy y (opposite of X)
Dim affiliationDescMatrix As String             'sheet that associates terms to affiliations

   Dim testBandA As Integer             'the test variables are used to determine if
   Dim testBandB As Integer             'there are records in a band
   Dim testBandC As Integer             'the variable stores the number of affilations
   Dim testBandD As Integer             'that are in each band



'CONSTANTS
   Private Const HYP3_FIT As Integer = 0
   Private Const EXP3_FIT As Integer = 1
   Private Const POW2_FIT As Integer = 2

Sub a_Test_Run() 'controlled the subs that are ran..

   testBandA = 1
   testBandB = 1
   testBandC = 1
   testBandD = 1


'sheet name constants
   descriptorMatrixSheet = "descriptor_data_X"
   descriptorMatrixSheetY = "descriptor_data_Y"
   worldEntropySheet = "World_Cumulative_Entropy_X"
   worldEntropySheetY = "World_Cumulative_Entropy_Y"
   affiliationDescMatrix = "descriptor_matrix_affil"

'subs
```

```
    Call instancesSummary 'used to check if the world instances match up with the # of instances of
                    ' the sum of the bands (a + b + c + d)

End Sub



'--------------------------------------------------------------------------
'  Sub: DistributeAffiliations
'  Author: Matt Behnke
'  Created: 11/5/01
'  Description: The sub routine that calls all the sub routines
'  inputs:
'
'  Outputs:
'--------------------------------------------------------------------------
Sub DistributeAffiliations()

    testBandA = 0
    testBandB = 0
    testBandC = 0
    testBandD = 0

    currFilename = Application.ActiveWorkbook.Name
    dataSheet = ActiveSheet.Name

'sheet name constants
    descriptorMatrixSheet = "descriptor_data_X"
    descriptorMatrixSheetY = "descriptor_data_Y"
    worldEntropySheet = "World_Cumulative_Entropy_X"
    worldEntropySheetY = "World_Cumulative_Entropy_Y"
    affiliationDescMatrix = "descriptor_matrix_affil"

    technologyName = InputBox("Enter the name of the technology.")
    stepInterval = InputBox("Enter the time between time steps")


    Call CopyMathCadObj

'put the cumulative values on the sheets:
    Call CalcCumulative(dataSheet)  'datasheet has the num records each affilation produced over time
    Call CalcCumulative(descriptorMatrixSheet)
    Call CalcCumulative(affiliationDescMatrix)
    Call CalcCumulative("Affiliation_authors")

'determine the num of records in each band
    Call AffiliationDistribution
'use the summary sheet created by Affiliation_Distribution to graph the distributions of each band:
    Call CopyDistributionGraph

'create descriptor data y sheet from descriptor data x sheet:
    Call CreateDescriptorDataY("descriptor_data_X", "descriptor_data_Y")

'compute world entropy sheets x and y (input, output)
    Call ComputeEntropy("descriptor_data_X", "World_Cumulative_Entropy_X", 1)
```

```
   Call ComputeEntropy("descriptor_data_Y", "World_Cumulative_Entropy_Y", 2)

'fills the band stats of the world:

   Call FillBandStats("World")

'compute nu and psi for the world:
   Call v_calc_v_psi_sheet("World")

'-----------------BANDS-------------------------------------

'fill the band with the affiliations and their number of publications that fit the number of
'publications range for that band determined by Affiliation_Distribution:

  testBandA = FillBand("A_Band")
'fill band stats:
   If testBandA > 0 Then
      Call FillBandStats("A_Band")
'create the matrix of affiliation with author instances
      Call FillBandAuthors("A_Band")
'calculate nu and psi:
      Call v_calc_v_psi_sheet("A_Band")
'determine the matrix of terms and the number of instances for the band
      Call FillBandTerms("A_Band")
'compute the entropy of the terms in the band
      Call FillBandTermsEntropy("A_Band")
'create a summaty of band.. num of publications, authors, terms, entropy:
      Call affiliationBandSummary("A_Band")
   End If


'
' put in a test for each band & store the results, then re-run the stuff
'
'
'


   testBandB = FillBand("B_Band")
   If testBandB > 0 Then
      Call FillBandStats("B_Band")
      Call FillBandAuthors("B_Band")
      Call v_calc_v_psi_sheet("B_Band")
      Call FillBandTerms("B_Band")
      Call FillBandTermsEntropy("B_Band")
      Call affiliationBandSummary("B_Band")
   End If

   testBandC = FillBand("C_Band")
   If testBandC > 0 Then
      Call FillBandStats("C_Band")
      Call FillBandAuthors("C_Band")
      Call v_calc_v_psi_sheet("C_Band")
      Call FillBandTerms("C_Band")
      Call FillBandTermsEntropy("C_Band")
      Call affiliationBandSummary("C_Band")
   End If
```

```
    testBandD = FillBand("D_Band")
    If testBandD > 0 Then
        Call FillBandStats("D_Band")
        Call FillBandAuthors("D_Band")
        Call v_calc_v_psi_sheet("D_Band")
        Call FillBandTerms("D_Band")
        Call FillBandTermsEntropy("D_Band")
        Call affiliationBandSummary("D_Band")
    End If


    Call instancesSummary 'used to check if the world instances match up with the # of instances of
                    ' the sum of the bands (a + b + c + d)
    Call entropySummary  'for the world
    Call affiliationSummary 'for the world
    Call affiliationSummaryPart2 'copies graphs and computes
    Call affiliationSummaryPart3 'temp and pressure...

    Call CopyABCDGraph 'copy the abcd learning curve graphs
    Call fillMonthsRowTrigger

    If testBandA > 0 Then
    Call CopyBandSummaryGraphs("A_Band") 'entropy summary graphs
    End If
    If testBandB > 0 Then
        Call CopyBandSummaryGraphs("B_Band")
    End If
    If testBandC > 0 Then
        Call CopyBandSummaryGraphs("C_Band")
    End If
    If testBandD > 0 Then
        Call CopyBandSummaryGraphs("D_Band")
    End If
    Call CopyBandSummaryGraphs("World")
End Sub

'--------------------------------------------------------------------------
'  Sub: AffliationDistribution
'  Author: Matt Behnke
'  Created: 11/5/01
'  Description: figures out the division of bands, and the number of affiliations per band
'  inputs:
'
'  Outputs:
'--------------------------------------------------------------------------
Sub AffiliationDistribution()

    numRows = CountRows(dataSheet, 1)

    Sheets.Add
    sheetName = "Distribution"
    ActiveSheet.Name = sheetName
```

```
    Sheets(sheetName).Cells(1, 1) = "Statistics"

    Sheets(sheetName).Cells(2, 1).FormulaR1C1 = "Mean"
    Sheets(sheetName).Cells(2, 2).Formula = "=AVERAGE('" & dataSheet & "'!A2:A" & numRows & ")"

    Sheets(sheetName).Cells(3, 1) = "Stdev"
    Sheets(sheetName).Cells(3, 2).Formula = "=STDEV('" & dataSheet & "'!A2:A" & numRows & ")"

    Sheets(sheetName).Cells(4, 1) = "Sum"
    Sheets(sheetName).Cells(4, 2).Formula = "=SUM('" & dataSheet & "'!A2:A" & numRows & ")"

    Sheets(sheetName).Cells(5, 1) = "Count"
    Sheets(sheetName).Cells(5, 2).Formula = numRows - 1

    Sheets(sheetName).Cells(2, 5).Formula = "Calculate Bands"
    Sheets(sheetName).Cells(3, 5).Formula = "Band_D"
    Sheets(sheetName).Cells(3, 6).Formula = "Band_C"
    Sheets(sheetName).Cells(3, 7).Formula = "Band_B"
    Sheets(sheetName).Cells(3, 8).Formula = "Band_A"
    Sheets(sheetName).Cells(4, 4).Formula = "from"
    Sheets(sheetName).Cells(5, 4).Formula = "to"

    Sheets(sheetName).Cells(4, 5).Formula = "0"                  'Band D from
    Sheets(sheetName).Cells(5, 5) = "=ROUND(B2+B3,3)"            'band d to

    Sheets(sheetName).Cells(4, 6) = "=ROUND(B2+B3,3)"            'band c from
    Sheets(sheetName).Cells(5, 6) = "=ROUND(B2+B3*2,3)"          'band c to

    Sheets(sheetName).Cells(4, 7) = "=ROUND(B2+B3*2,3)"          'band b from
    Sheets(sheetName).Cells(5, 7) = "=ROUND(B2+B3*3,3)"          'band b to

    Sheets(sheetName).Cells(4, 8) = "=ROUND(B2+B3*3,3)"          'band a from

  'bin labels
    Sheets(sheetName).Cells(7, 1) = "Bin"
    Sheets(sheetName).Cells(7, 2) = "Frequency"

    counter = 1
    For i = 1 To Round(Sheets(sheetName).Cells(5, 5).Value, 0) 'get bin values for band A
        Sheets(sheetName).Cells(7 + i, 1) = i
        Sheets(sheetName).Cells(7 + i, 2) = "=COUNTIF('" & dataSheet & "'!A2:A" & numRows & ", ""=" &
i & """)"
        counter = counter + 1
    Next i

    Sheets(sheetName).Cells(7 + counter, 1) = Sheets(sheetName).Cells(5, 6) 'put in next bin (band c end)
    Sheets(sheetName).Cells(1, 9).Formula = "=COUNTIF('" & dataSheet & "'!A2:A" & numRows & ",
""<" & Sheets(sheetName).Cells(5, 6) & """)"
    Sheets(sheetName).Cells(1, 10).Formula = "=COUNTIF('" & dataSheet & "'!A2:A" & numRows & ",
""<" & Sheets(sheetName).Cells(4, 6) & """)"
    Sheets(sheetName).Cells(7 + counter, 2) = Abs(Sheets(sheetName).Cells(1, 9) -
Sheets(sheetName).Cells(1, 10))
        counter = counter + 1

    Sheets(sheetName).Cells(7 + counter, 1) = Sheets(sheetName).Cells(5, 7) 'band b end
```

```
   Sheets(sheetName).Cells(1, 9).Formula = "=COUNTIF('" & dataSheet & "'!A2:A" & numRows & ",
""<" & Sheets(sheetName).Cells(5, 7) & """)"
   Sheets(sheetName).Cells(1, 10).Formula = "=COUNTIF('" & dataSheet & "'!A2:A" & numRows & ",
""<" & Sheets(sheetName).Cells(4, 7) & """)"
   Sheets(sheetName).Cells(7 + counter, 2) = Abs(Sheets(sheetName).Cells(1, 9) -
Sheets(sheetName).Cells(1, 10))
      counter = counter + 1

   exitIf = False
   If Sheets(sheetName).Cells(5, 7) < 15 Then           'add more bins 15-30...
      Sheets(sheetName).Cells(7 + counter, 1) = "15"
      Sheets(sheetName).Cells(1, 9).Formula = "=COUNTIF('" & dataSheet & "'!A2:A" & numRows & ",
""<= 15"")"
      Sheets(sheetName).Cells(1, 10).Formula = "=COUNTIF('" & dataSheet & "'!A2:A" & numRows & ",
""<" & Sheets(sheetName).Cells(4, 8) & """)"
      Sheets(sheetName).Cells(7 + counter, 2) = Abs(Sheets(sheetName).Cells(1, 9) -
Sheets(sheetName).Cells(1, 10))
      If Sheets(sheetName).Cells(7 + counter, 2) = 0 Then
         Sheets(sheetName).Cells(7 + counter, 1) = "> " & Sheets(sheetName).Cells(4, 8)
         Sheets(sheetName).Cells(7 + counter, 2).Formula = "=COUNTIF('" & dataSheet & "'!A2:A" &
numRows & ", "">=" & Sheets(sheetName).Cells(4, 8) & """)"
         exitIf = True
      End If
      counter = counter + 1

      If exitIf = False Then
         Sheets(sheetName).Cells(7 + counter, 1) = "20"
         Sheets(sheetName).Cells(1, 9).Formula = "=COUNTIF('" & dataSheet & "'!A2:A" & numRows &
", ""<= 20"")"
         Sheets(sheetName).Cells(1, 10).Formula = "=COUNTIF('" & dataSheet & "'!A2:A" & numRows &
", ""< 16"")"
         Sheets(sheetName).Cells(7 + counter, 2) = Abs(Sheets(sheetName).Cells(1, 9) -
Sheets(sheetName).Cells(1, 10))
         If Sheets(sheetName).Cells(7 + counter, 2) = 0 Then
            Sheets(sheetName).Cells(7 + counter, 1) = "> 15"
            Sheets(sheetName).Cells(7 + counter, 2).Formula = "=COUNTIF('" & dataSheet & "'!A2:A" &
numRows & ", "">= 15"")"
            exitIf = True
         End If
      End If ' exitif
      counter = counter + 1

      If exitIf = False Then
         Sheets(sheetName).Cells(7 + counter, 1) = "25"
         Sheets(sheetName).Cells(1, 9).Formula = "=COUNTIF('" & dataSheet & "'!A2:A" & numRows &
", ""<= 25"")"
         Sheets(sheetName).Cells(1, 10).Formula = "=COUNTIF('" & dataSheet & "'!A2:A" & numRows &
", ""< 21"")"
         Sheets(sheetName).Cells(7 + counter, 2) = Abs(Sheets(sheetName).Cells(1, 9) -
Sheets(sheetName).Cells(1, 10))
         If Sheets(sheetName).Cells(7 + counter, 2) = 0 Then
            Sheets(sheetName).Cells(7 + counter, 1) = "> 20"
            Sheets(sheetName).Cells(7 + counter, 2).Formula = "=COUNTIF('" & dataSheet & "'!A2:A" &
numRows & ", "">= 20"")"
            exitIf = True
         End If
```

```
        End If ' exitif
        counter = counter + 1

        If exitIf = False Then
            Sheets(sheetName).Cells(7 + counter, 1) = "30"
            Sheets(sheetName).Cells(1, 9).Formula = "=COUNTIF('" & dataSheet & "'!A2:A" & numRows &
", ""<= 30"")"
            Sheets(sheetName).Cells(1, 10).Formula = "=COUNTIF('" & dataSheet & "'!A2:A" & numRows &
", ""< 26"")"
            Sheets(sheetName).Cells(7 + counter, 2) = Abs(Sheets(sheetName).Cells(1, 9) -
Sheets(sheetName).Cells(1, 10))
            If Sheets(sheetName).Cells(7 + counter, 2) = 0 Then
                Sheets(sheetName).Cells(7 + counter, 1) = "> 25"
                Sheets(sheetName).Cells(7 + counter, 2).Formula = "=COUNTIF('" & dataSheet & "'!A2:A" &
numRows & ", ""> 25"")"
                exitIf = True
            End If
        End If ' exitif
        counter = counter + 1

        If exitIf = False Then
            Sheets(sheetName).Cells(7 + counter, 1) = "> 30"
            Sheets(sheetName).Cells(7 + counter, 2) = "=COUNTIF('" & dataSheet & "'!A2:A" & numRows &
", ""> 30"")"
        End If

    Else
        If exitIf = False Then
            Sheets(sheetName).Cells(7 + counter, 1) = "20"
            Sheets(sheetName).Cells(1, 9).Formula = "=COUNTIF('" & dataSheet & "'!A2:A" & numRows &
", ""<= 20"")"
            Sheets(sheetName).Cells(1, 10).Formula = "=COUNTIF('" & dataSheet & "'!A2:A" & numRows &
", ""<= 15"")"
            Sheets(sheetName).Cells(7 + counter, 2) = Abs(Sheets(sheetName).Cells(1, 9) -
Sheets(sheetName).Cells(1, 10))
            If Sheets(sheetName).Cells(7 + counter, 2) = 0 Then
                Sheets(sheetName).Cells(7 + counter, 1) = "> 15"
                Sheets(sheetName).Cells(7 + counter, 2).Formula = "=COUNTIF('" & dataSheet & "'!A2:A" &
numRows & ", ""> 15"")"
                exitIf = True
            End If
        End If ' exitif
        counter = counter + 1

        If exitIf = False Then
            Sheets(sheetName).Cells(7 + counter, 1) = "25"
            Sheets(sheetName).Cells(1, 9).Formula = "=COUNTIF('" & dataSheet & "'!A2:A" & numRows &
", ""<= 25"")"
            Sheets(sheetName).Cells(1, 10).Formula = "=COUNTIF('" & dataSheet & "'!A2:A" & numRows &
", ""<= 20"")"
            Sheets(sheetName).Cells(7 + counter, 2) = Abs(Sheets(sheetName).Cells(1, 9) -
Sheets(sheetName).Cells(1, 10))
            If Sheets(sheetName).Cells(7 + counter, 2) = 0 Then
                Sheets(sheetName).Cells(7 + counter, 1) = "> 20"
                Sheets(sheetName).Cells(7 + counter, 2).Formula = "=COUNTIF('" & dataSheet & "'!A2:A" &
numRows & ", ""> 20"")"
```

```
            exitIf = True
         End If
      End If ' exitif
      counter = counter + 1

      If exitIf = False Then
         Sheets(sheetName).Cells(7 + counter, 1) = "30"
         Sheets(sheetName).Cells(1, 9).Formula = "=COUNTIF('" & dataSheet & "'!A2:A" & numRows &
", ""<= 30"")"
         Sheets(sheetName).Cells(1, 10).Formula = "=COUNTIF('" & dataSheet & "'!A2:A" & numRows &
", ""<= 25"")"
         Sheets(sheetName).Cells(7 + counter, 2) = Abs(Sheets(sheetName).Cells(1, 9) -
Sheets(sheetName).Cells(1, 10))
         If Sheets(sheetName).Cells(7 + counter, 2) = 0 Then
            Sheets(sheetName).Cells(7 + counter, 1) = "> 25"
            Sheets(sheetName).Cells(7 + counter, 2).Formula = "=COUNTIF('" & dataSheet & "'!A2:A" &
numRows & ", "">= 25"")"
            exitIf = True
         End If
      End If ' exitif
      counter = counter + 1

      If exitIf = False Then
         Sheets(sheetName).Cells(7 + counter, 1) = "> 30"
         Sheets(sheetName).Cells(7 + counter, 2) = "=COUNTIF('" & dataSheet & "'!A2:A" & numRows &
", ""> 30"")"
      End If
   End If

Call formatSheetForPrint

End Sub
'---------------------------------------------------------------------------
'  Sub: CopyMathCadObj
'  Author: Matt Behnke
'  Created: 12/5/01
'  Description: copies the mathcad onject, for running a curve fit..
'  inputs:
'
'  Outputs:
'---------------------------------------------------------------------------
Sub CopyMathCadObj()

   Windows("AffiliationMacro.xls").Activate
   Sheets("Mathcad").Select
   Sheets("Mathcad").Copy Before:=Workbooks(currFilename).Sheets(dataSheet)
'   If ActiveSheet.Name = "Mathcad" Then
'      ActiveSheet.Name = "Mathcad_" & band
'   Else
'      MsgBox ("Mathcad sheet rename failed")
'   End If

End Sub

'---------------------------------------------------------------------------
'  Sub: CopyDistributionGraph
```

```
'  Author: Matt Behnke
'  Created: 11/5/01
'  Description: copies the distribution graph from the macro sheet into the data spreadsheet
'  inputs:
'
'  Outputs:
'--------------------------------------------------------------------------
Sub CopyDistributionGraph()

Application.DisplayAlerts = False

   numSheets = Sheets.Count

   Windows("AffiliationMacro.xls").Activate
   Sheets("Affiliation Distribution Sample").Select
   Sheets("Affiliation Distribution Sample").Copy After:=Workbooks(currFilename).Sheets(numSheets)
   Sheets("Affiliation Distribution Sample").Copy After:=Workbooks(currFilename).Sheets(numSheets)

   ActiveChart.SeriesCollection(1).Select

   ActiveChart.SeriesCollection(1).XValues = "=Distribution!R8C1:R18C1"
   ActiveChart.SeriesCollection(1).Values = "=Distribution!R8C2:R19C2"
   ActiveChart.ChartTitle.Characters.Text = "Productivity Distribution" & Chr(10) _
   & technologyName & " (" & stepInterval & ")"

Application.DisplayAlerts = True

End Sub

'--------------------------------------------------------------------------
'  Sub: ComputeEntropy
'  Author: Matt Behnke
'  Created: 1/28/02
'  Description: Computes the cumulative entropy using the supplied datasheets
'               note number of instances must begin at row 2, column 4..
'  inputs: datasheet - matrix of the descriptorData.. Y-axis is the terms, X-axis is timesteps, v is # of
instances
'          time  1,  2,  3,  4 .....
'          term1 v   v   v
'          term2 v
'         outSheet: name of the sheet that contains the computed entropy.
'          theType: 1) s(x|y), 2) s(y|x)
'  Outputs:
'--------------------------------------------------------------------------
Sub ComputeEntropy(ByVal dataSheet As String, ByVal outSheet As String, ByVal theType As Integer)

   numCols = CountCols(dataSheet, 1)
   numRows = CountRows(dataSheet, 1)

   Sheets.Add
   'Sheets(Sheets.Count).Select
   ActiveSheet.Name = outSheet
   Worksheets(outSheet).Move After:=Worksheets(dataSheet)


   For i = 1 To numCols
```

```
      If i >= 4 And theType = 1 Then
         TotalNumInstances = Sheets(dataSheet).Cells(numRows + 1, i)
      ElseIf i >= 4 And theType = 2 Then
         TotalNumInstances = Sheets(dataSheet).Cells(numRows + 1, 4)
      End If

      For j = 1 To numRows

         If i >= 4 And j >= 2 Then
            numInstances = Sheets(dataSheet).Cells(j, i)
            If Not numInstances = "" And numInstances > 0 Then ' Then

               entropy = -numInstances / TotalNumInstances * (Log(numInstances / TotalNumInstances) /
Log(2))

               Sheets(outSheet).Cells(j, i) = entropy
            End If
         If j = numRows Then 'put in sum of entropy
            Sheets(outSheet).Cells(j + 1, i) = "=SUM(" & col(i) & "2:" & col(i) & numRows & ")"
         End If
         Else 'copy terms, count, first pub date
            Sheets(outSheet).Cells(j, i).Value = Sheets(dataSheet).Cells(j, i).Value
         End If
      Next j
   Next i

End Sub

'---------------------------------------------------------------------------
'  Sub: CreateDescriptorDataY
'  Author: Matt Behnke
'  Created: 1/28/02
'  Description: Takes the supplied descriptor data sheet and creates the Y part of the (X,Y) world
'               as x increases y decreases.. a value decreases on the y sheet when a value increases on the y sheet
'  inputs: datasheet - matrix of the descriptorData.. Y-axis is the terms, X-axis is timesteps, v is # of
instances
'          time  1,  2,  3,  4 .....
'          term1 v   v   v
'          term2 v
'         outSheet: name of the sheet that contains DescriptorDataY
'  Outputs:
'---------------------------------------------------------------------------

Sub CreateDescriptorDataY(ByVal dataSheet As String, ByVal outSheet As String)

   numCols = CountCols(dataSheet, 1)
   numRows = CountRows(dataSheet, 1)

   Worksheets(dataSheet).Copy After:=Worksheets(dataSheet)

   Sheets(dataSheet & " (2)").Select
   ActiveSheet.Name = outSheet

   For i = 2 To numRows

      For j = 4 To numCols
```

```
            numTotalInstances = Sheets(outSheet).Cells(i, 2)
            numInstances = Sheets(outSheet).Cells(i, j)

            If j = 4 And Sheets(outSheet).Cells(i, j) > 0 Then 'places the initial value at the end..
               lastColumn = Sheets(outSheet).Cells(i, j)
            End If

            Sheets(outSheet).Cells(i, j) = numTotalInstances - numInstances

            If j = numCols And lastColumn > 0 Then
               Sheets(outSheet).Cells(i, j) = lastColumn 'places the value of first column x into last coln Y.
            End If

            If i = numRows Then 'put in sum
               Sheets(outSheet).Cells(i + 1, j) = "=SUM(" & col(j) & "2:" & col(j) & numRows & ")"
            End If
         Next j
         lastColumn = 0
      Next i

End Sub 'CreateDescriptorDataY


Sub computeEntropyTest()
'IT WORKS

'Call ComputeEntropy("descriptor_data_X", "World_Cumulative_Entropy_X", 1)
Call ComputeEntropy("descriptor_data_Y", "World_Cumulative_Entropy_Y", 2)
'Call CreateDescriptorDataY("descriptor_data_X", "descriptor_data_Y")
End Sub

'---------------------------------------------------------------------------
'  Sub: FillBand
'  Author: Matt Behnke
'  Created: 11/7/01
'  Description: fills in a bands distribution by copying a row from the list of all the affiliations (datasheet)
'  inputs: band name
'
'  Outputs: num of rows copied
'---------------------------------------------------------------------------
Function FillBand(ByVal band As String) As Integer

    numRowsCopied = 0

   Sheets.Add After:=Worksheets(Worksheets.Count)
   currsheetName = ActiveSheet.Name
    numRows = CountRows(dataSheet, 1)
    'Sheets(Worksheets.Count).Select

    Columns("C:C").ColumnWidth = 62.43

    Select Case band
       Case "A_Band"
          bandFrom = Sheets("Distribution").Cells(4, 8)
          bandTo = 32500
```

```
        Case "B_Band"
            bandFrom = Sheets("Distribution").Cells(4, 7)
            bandTo = Sheets("Distribution").Cells(5, 7)
        Case "C_Band"
            bandFrom = Sheets("Distribution").Cells(4, 6)
            bandTo = Sheets("Distribution").Cells(5, 6)
        Case "D_Band"
            bandFrom = Sheets("Distribution").Cells(4, 5)
            bandTo = Sheets("Distribution").Cells(5, 5)
    End Select

    Sheets("" & dataSheet & "").Select
    Rows("1:1").Select
    Selection.Copy
    Sheets(currsheetName).Select
    Rows("1:1").Select
    ActiveSheet.Paste

    counter = 2
    For i = 2 To numRows 'copy rows from datasheet into band
        If Sheets(dataSheet).Cells(i, 1) >= bandFrom And Sheets(dataSheet).Cells(i, 1) <= bandTo Then
            Sheets(dataSheet).Select
            Rows(i & ":" & i).Select
            Selection.Copy
            Sheets(currsheetName).Select
            Rows(counter & ":" & counter).Select
            ActiveSheet.Paste
            counter = counter + 1
            numRowsCopied = numRowsCopied + 1
        End If
    Next i

    numRowsInBand = CountRows(currsheetName, 1)
    numColumns = CountCols(currsheetName, 1) 'num time steps

    Cells(numRowsInBand + 1, 3) = "Count"
    Cells(numRowsInBand + 2, 3) = "Mean"
    Cells(numRowsInBand + 3, 3) = "Std Dev"
    Cells(numRowsInBand + 4, 3) = "Sum"

    For i = 4 To numColumns 'put in the mean and std deviation for each time step
'put in zeros if nothing there
'       For j = 2 To numRowsInBand
'           If i = 4 Then
'               If Cells(j, i) > 0 Then
'               Else
'                   Cells(j, i) = 0
'               End If
'           Else
'               Cells(j, i) = Cells(j, i) + Cells(j, i - 1)
'           End If
'       Next j

'dont put in zeros if nothing there
'       For j = 2 To numRowsInBand
'           If (Cells(j, i) > 0 And i > 4) Or (i > 4 And Cells(j, i - 1) > 0) Then
```

```
'               Cells(j, i) = Cells(j, i) + Cells(j, i - 1)
'           End If
'       Next j

       Cells(numRowsInBand + 4, i).Formula = "=Sum(" & col(i) & "2:" & col(i) & numRowsInBand & ")"
       Cells(numRowsInBand + 1, i).Formula = "=Countif(" & col(i) & "2:" & col(i) & numRowsInBand &
", ""0"")"
       If Cells(numRowsInBand + 1, i) > 0 Then
          Cells(numRowsInBand + 2, i).Formula = "=AVERAGE(" & col(i) & "2:" & col(i) &
numRowsInBand & ")"
          If Cells(numRowsInBand + 1, i) > 1 Then 'more than one so comput std deviation
             Cells(numRowsInBand + 3, i).Formula = "=STDEV(" & col(i) & "2:" & col(i) &
numRowsInBand & ")"
          End If
       End If
   Next i

   ActiveSheet.Name = "Affiliation_Cum_Dist_" & band
   'Call formatSheetForPrint
   FillBand = numRowsCopied

End Function

'---------------------------------------------------------------------------
'  Sub: FillBandStats
'  Author: Matt Behnke
'  Created: 11/7/01
'  revised: 12/3/01
'  Description: creates a band's statistics sheet
'  inputs: band name
'
'  Outputs:
'---------------------------------------------------------------------------
Sub FillBandStats(ByVal band As String)

   Dim data As Variant

   If band = "World" Then
      source = dataSheet
   Else
      source = "Affiliation_Cum_Dist_" & band
   End If

   numRowsInBand = CountRows(source, 1)
   numTimeStepsInBand = CountCols(source, 1) - 3

   Sheets.Add
   sheetName = band & "_Stats"
   ActiveSheet.Name = sheetName "" & band & "_Stats"

   Sheets(sheetName).Move After:=Sheets(source)

  ' Sheets(Worksheets.Count).Select
   'Columns("C:C").ColumnWidth = 62.43

  Sheets(sheetName).Cells(5, 1) = " "
```

```
  Sheets(sheetName).Cells(6, 1) = " "
  Sheets(sheetName).Cells(7, 1) = " "
  Sheets(sheetName).Cells(8, 1) = " "
  Sheets(sheetName).Cells(9, 1) = " "
  Sheets(sheetName).Cells(11, 1) = " "
  Sheets(sheetName).Cells(12, 1) = " "


  Sheets(sheetName).Cells(1, 1) = "Curve fit y(t)  y(t) = bt^m"
  Sheets(sheetName).Cells(3, 1) = "b"
  Sheets(sheetName).Cells(4, 1) = "m"

  Sheets(sheetName).Cells(8, 3) = "Total Production"
  Sheets(sheetName).Cells(8, 6) = "Production/Step (on Average)"
  Sheets(sheetName).Cells(8, 11) = "Calculated Production/Step)"

  Sheets(sheetName).Cells(10, 1) = "Time Step"
  Sheets(sheetName).Cells(10, 2) = "Step Name"

  Sheets(sheetName).Cells(10, 3) = "Mean"
  Sheets(sheetName).Cells(10, 4) = "Std Deviation"
  Sheets(sheetName).Cells(10, 5) = "'+ 3 sigma"
 'average per step
  Sheets(sheetName).Cells(10, 6) = "Mean"
  Sheets(sheetName).Cells(10, 7) = "Std Deviation"
  Sheets(sheetName).Cells(10, 8) = "'+ 3 sigma"

  Sheets(sheetName).Cells(10, 9) = "Total Prod"
  Sheets(sheetName).Cells(10, 10) = "kappa"
  Sheets(sheetName).Cells(10, 11) = "kappa/2"
  Sheets(sheetName).Cells(10, 12) = "r value"

  Sheets(sheetName).Cells(10, 13) = "Mean"
  Sheets(sheetName).Cells(10, 14) = "R^2"
  Sheets(sheetName).Cells(10, 15) = "'+ 3 sigma"


  Sheets(sheetName).Cells(13, 1) = "0" 'time step zero

   For i = 1 To numTimeStepsInBand
      Sheets("" & band & "_Stats").Select
     Sheets(sheetName).Cells(13 + i, 1) = i 'step number
     Sheets(sheetName).Cells(13 + i, 2) = Sheets(source).Cells(1, i + 3) 'step name
    'Total Production
     Sheets(sheetName).Cells(13 + i, 3) = 0.000001 + Sheets(source).Cells(numRowsInBand + 2, i +
3).Value     'mean
     Sheets(sheetName).Cells(13 + i, 4) = 0.000001 + Sheets(source).Cells(numRowsInBand + 3, i +
3).Value      'stdev
     Sheets(sheetName).Cells(13 + i, 5) = "=C" & 13 + i & "+ 3*D" & 13 + i 'sheets(sheetName).Cells(13 +
i, 4) 'mean + 3std
    'Production per step on avg..
      If i = 1 Then
        'Cells(13 + i, 6) =sheets(sheetName).Cells(13 + i, 3) /sheets(sheetName).Cells(13 + i, 1) 'mean
        Sheets(sheetName).Cells(13 + i, 6) = "=C" & 13 + i        'Cells(13 + i, 3)
        Sheets(sheetName).Cells(13 + i, 7) = "=D" & 13 + i     'Cells(13 + i, 4) 'stdev
        Sheets(sheetName).Cells(13 + i, 8) = "=E" & 13 + i 'sheets(sheetName).Cells(13 + i, 5) 'mean * 3std
```

```
        Else
          Sheets(sheetName).Cells(13 + i, 6) = "=C" & 13 + i & " - C" & 12 + i  'Cells(13 + i, 3) -
sheets(sheetName).Cells(12 + i, 3)
          Sheets(sheetName).Cells(13 + i, 7) = "=D" & 13 + i & " - D" & 12 + i  'Cells(13 + i, 4) -
sheets(sheetName).Cells(12 + i, 4) 'stdev
          Sheets(sheetName).Cells(13 + i, 8) = "=E" & 13 + i & " - E" & 12 + i  'Cells(13 + i, 5) -
sheets(sheetName).Cells(12 + i, 5) 'mean + 3std
        End If

        If i = numTimeStepsInBand Then  'put in average
          Sheets(sheetName).Cells(14 + i, 8) = "=AVERAGE(H14:H" & i + 13 & ")"
        End If

        If i = 1 Then
          Sheets(sheetName).Cells(13 + i, 9) = "=H" & 13 + i      'Cells(13 + i, 8)
        Else
          Sheets(sheetName).Cells(13 + i, 9) = "=H" & 13 + i & " + H" & 12 + i  'Cells(13 + i, 8)
+sheets(sheetName).Cells(12 + i, 9)
        End If

       Sheets(sheetName).Cells(13 + i, 10) = "=H" & 14 + numTimeStepsInBand 'avg of mean*3std
    Next i

    'ActiveSheet.Name = "" & band & "_Stats"
    Call copyStatGraphs(numTimeStepsInBand, band, "" & band & "_Stats")

    Sheets("" & band & "_Stats").Select

   'get formula of trendline from entropy power trend graph
     trendEq = Sheets(sheetName).Cells(2, 1)
    Sheets(sheetName).Cells(3, 2) = firstPartTrendEq(trendEq)
    Sheets(sheetName).Cells(4, 2) = secondPartTrendEq(trendEq)

     'get kappa, r. p

    Sheets(sheetName).Cells(3, 3) = "kappa"  'headers
    Sheets(sheetName).Cells(4, 3) = "r"
    Sheets(sheetName).Cells(5, 3) = "p"
    Sheets(sheetName).Cells(6, 3) = "1-Sum(r^2)"

     j = 14 'get the start row
'    While Not (i > 0#)   'if the first time step's mean is zero find the step that doesnt have 0
'       i =sheets(sheetName).Cells(j, 3).Value
'       If Not (i > 0) Then
'          j = j + 1
'       End If
'    Wend

    'perform linear interpolation on columns C and D
     'dataSheet = band & "_Stats"
     Call FillInMissingData(sheetName, 3, 14, numTimeStepsInBand + 13)
     Call FillInMissingData(sheetName, 4, 14, numTimeStepsInBand + 13)

     Sheets("" & band & "_Stats").Select

     'j = j + 1 'add one to the starting row to not include the first time step....
```

```
   numRowsToUse = numTimeStepsInBand - (j - 14)
   data = Update_Mathcad_Band_Stats("Mathcad", ActiveSheet.Name, "C" & j, "F" & j, numRowsToUse -
1, 0, 0.001)

   kappa = data(1)
   r = data(2)
   p = data(3)
   r2a = data(4)

   'put on sheet
  Sheets(sheetName).Cells(3, 4) = Round(kappa, 4)
  Sheets(sheetName).Cells(4, 4) = Round(r, 4)
  Sheets(sheetName).Cells(5, 4) = Round(p, 4)
  Sheets(sheetName).Cells(6, 5) = Round(r2a, 4)

 'calculate prediticded means for -1, -2 under total
   'Cells(11, 3).Formula = "=-$B$3*-A11^$B$4"   'not needed
   'Cells(12, 3).Formula = "=-$B$3*-A12^$B$4"
   'Cells(13, 3) = 0


   For i = 1 To numTimeStepsInBand
     'fill in kappa, kappa/2, rvalue
      Sheets(sheetName).Cells(13 + i, 10) = "=$D$3"
      Sheets(sheetName).Cells(13 + i, 11) = "=$D$3 / 2"
      Sheets(sheetName).Cells(13 + i, 12) = "=$D$4"

     'fill in calculated prod per step
      Sheets(sheetName).Cells(13 + i, 13).Formula = "=$D$3*(C" & 13 + i & "+$D$5)/(C" & 13 + i &
"+$D$4+$D$5)" 'mean
      Sheets(sheetName).Cells(13 + i, 14) = "=(M" & 13 + i & "-F" & 13 + i & ")*(M" & 13 + i & "-F" &
13 + i & ")" 'R^2
       sumRSquared = Sheets(sheetName).Cells(13 + i, 14) + sumRSquared
      Sheets(sheetName).Cells(13 + i, 15) = "=$D$3*(E" & 13 + i & "+$D$5)/(E" & 13 + i &
"+$D$4+$D$5)"

       If i = numTimeStepsInBand Then  'put in average R^2 -------- REMOVE AFTER dbl Checking
values.....
         Sheets(sheetName).Cells(15 + i, 10) = "Sum(R^2)"
         Sheets(sheetName).Cells(15 + i, 12) = "=Sum(N14:N" & i + 13 & ")"
         Sheets(sheetName).Cells(16 + i, 12) = "=1-L" & i + 15
       End If

   Next i

   inverseRSquared = (1 - sumRSquared) 'from the sum of r squared
  Sheets(sheetName).Cells(6, 4) = Round(inverseRSquared, 4) '4decimal places

'   sheets(sheetName).Cells(11, 11).Formula = "=$D$3*(C11+$D$5)/(C11+$D$4+$D$5)"  -removed (-2, -
1, 0 time steps of calculated mean)
'   sheets(sheetName).Cells(12, 11).Formula = "=$D$3*(C12+$D$5)/(C12+$D$4+$D$5)"
'   sheets(sheetName).Cells(13, 11) = "=$D$3*(C13+$D$5)/(C13+$D$4+$D$5)"

   'Call formatSheetForPrint
   Call copyLearningCum(numTimeStepsInBand, band, "" & band & "_Stats") 'learning vs. cum
```

```
End Sub 'fill stats

'--------------------------------------------------------------------------
'  Sub: copyStatGraphs
'  Author: Matt Behnke
'  Created: 11/7/01
'  Description: copyies the affilaition statistics graphs
'  inputs:
'
'  Outputs:
'--------------------------------------------------------------------------
Sub copyStatGraphs(ByVal timeSteps As Integer, ByVal band As String, ByVal source As String)

Application.DisplayAlerts = False

    numSheets = Sheets.Count

    Windows("AffiliationMacro.xls").Activate
    Sheets("A_Band_Learning_Cap_per_k").Select
    Sheets("A_Band_Learning_Cap_per_k").Copy After:=Workbooks(currFilename).Sheets(numSheets)
    ActiveChart.SeriesCollection(1).Select

    j = 14
    While Not (i > 0#)   'if the first time step's mean is zero find the step that doesnt have 0
        i = Sheets(source).Cells(j, 3).Value
        If Not (i > 0) Then
            j = j + 1
        End If
    Wend

    ActiveChart.SeriesCollection(1).Values = "=" & source & "!R" & j & "C6:R" & timeSteps + 13 & "C6"

    ActiveChart.SeriesCollection(2).Values = "=" & source & "!R" & j + 1 & "C8:R" & timeSteps + 13 &
"C8"

    ActiveChart.ChartTitle.Characters.Text = "" & band & " Productivity Index (Cum over k)" & Chr(10) _
    & technologyName & " (" & stepInterval & ")"
    ActiveSheet.Name = "" & band & "_Learning_Cap_per_k"

    ActiveChart.SeriesCollection(1).ErrorBars.Select
    ExecuteExcel4Macro _
        "ERRORBAR.Y(2,5,""=" & source & "!R" & j & "C7:R" & timeSteps + 13 &
"C7"",""=A_Band_Stats!$F$" & j & ":$F$" & timeSteps + 13 & """)"

  'move legend and textbox
    ActiveChart.Legend.Select
    Selection.Left = 431
    Selection.Top = 341

  'copy second graph
    Windows("AffiliationMacro.xls").Activate
    Sheets("A_Band_Learning_Cum").Select
    Sheets("A_Band_Learning_Cum").Copy After:=Workbooks(currFilename).Sheets(numSheets)
    ActiveChart.SeriesCollection(1).Select
```

```
   ActiveChart.SeriesCollection(1).XValues = "=" & source & "!R" & j & "C1:R" & timeSteps + 13 &
"C1"
   ActiveChart.SeriesCollection(1).Values = "=" & source & "!R" & j & "C3:R" & timeSteps + 13 & "C3"

   ActiveChart.SeriesCollection(2).XValues = "=" & source & "!R" & j & "C1:R" & timeSteps + 13 &
"C1"
   ActiveChart.SeriesCollection(2).Values = "=" & source & "!R" & j & "C5:R" & timeSteps + 13 & "C5"


   ActiveChart.SeriesCollection(1).Select
  With ActiveChart.SeriesCollection(1).Trendlines(1)
   'put trendline equation onto stats sheet
    Worksheets(source).Cells(2, 1).Value = .DataLabel.Text
   .DisplayRSquared = True
  End With


   ActiveChart.ChartTitle.Characters.Text = "" & band & " Productivity In Pubs (Cum over k)" & Chr(10)
_
   & technologyName & " (" & stepInterval & ")"
   ActiveSheet.Name = "" & band & "_Learning_Cum"

Application.DisplayAlerts = True

End Sub

'***********
' copies the learning vs cumulative chart.....
'
'


Sub copyLearningCum(ByVal timeSteps As Integer, ByVal band As String, ByVal source As String)

Application.DisplayAlerts = False

   kappa = Sheets(source).Cells(3, 4)
   r = Sheets(source).Cells(4, 4)
   p = Sheets(source).Cells(5, 4)
   r2 = Sheets(source).Cells(6, 4)

  numSheets = Sheets.Count

 Windows("AffiliationMacro.xls").Activate
   Sheets("A_Band_Learning_Vs_Cum").Select
   Sheets("A_Band_Learning_Vs_Cum").Copy After:=Workbooks(currFilename _
      ).Sheets(numSheets)
   ActiveChart.PlotArea.Select

   j = 14
   While Not (i > 0#)   'if the first time step's mean is zero find the step that doesnt have 0
      i = Sheets(source).Cells(j, 3).Value
      If Not (i > 0) Then
         j = j + 1
      End If
   Wend
```

```
   ActiveChart.SeriesCollection(1).XValues = "=" & source & "!R" & j & "C3:R" & timeSteps + 13 &
"C3"
   ActiveChart.SeriesCollection(1).Values = "=" & source & "!R" & j & "C6:R" & timeSteps + 13 & "C6"

   ActiveChart.SeriesCollection(2).XValues = "=" & source & "!R" & j & "C3:R" & timeSteps + 13 &
"C3"
   ActiveChart.SeriesCollection(2).Values = "=" & source & "!R14C13:R" & timeSteps + 13 & "C13"

   'kappa
   ActiveChart.SeriesCollection(3).XValues = "=" & source & "!R" & j & "C5:R" & timeSteps + 13 &
"C5"
   ActiveChart.SeriesCollection(3).Values = "=" & source & "!R14C10:R" & timeSteps + 13 & "C10"

   '3sigma 3sigma
   ActiveChart.SeriesCollection(4).XValues = "=" & source & "!R" & j & "C5:R" & timeSteps + 13 &
"C5" 'E
   ActiveChart.SeriesCollection(4).Values = "=" & source & "!R14C8:R" & timeSteps + 13 & "C8" 'H

   'kappa/2
   ActiveChart.SeriesCollection(5).XValues = "=" & source & "!R" & j & "C5:R" & timeSteps + 13 &
"C5"
   ActiveChart.SeriesCollection(5).Values = "=" & source & "!R14C11:R" & timeSteps + 13 & "C11"

   'r-p
   ActiveChart.SeriesCollection(6).XValues = "=" & source & "!R" & j & "C12:R" & timeSteps + 13 &
"C12"
   ActiveChart.SeriesCollection(6).Values = "=" & source & "!R14C6:R" & timeSteps + 13 & "C6"

'    ActiveChart.SeriesCollection(1).ErrorBars.Select
 '   ExecuteExcel4Macro _
  '     "ERRORBAR.Y(2,5,""=" & source & "!R" & j & "C7:R" & timeSteps + 13 &
"C7"",""=A_Band_Stats!$F$" & j & ":$F$" & timeSteps + 13 & """)"

   ActiveChart.Shapes("Text Box 6").Select
   Selection.Characters.Text = "K= " & kappa & Chr(10) & "r= " & r & Chr(10) & "p= " & p & Chr(10) &
"" & Chr(10) & "R2= " & r2 & Chr(10) & "" & Chr(10) & ""

   ActiveChart.ChartTitle.Characters.Text = "Learning Curve -- " & band & " (Mean and Capacity)" &
Chr(10) _
   & technologyName & " (" & stepInterval & ")"
   ActiveSheet.Name = "" & band & "_Learning_Vs_Cum"
Application.DisplayAlerts = True

End Sub
'--------------------------------------------------------------------------
'  Sub: CopyABCDGraph
'  Author: Matt Behnke
'  Created: 11/7/01
'  Description: copies the ABCD band mean graph and changes the dataseries to point to the right data..
'  inputs:
'
'  Outputs:
'--------------------------------------------------------------------------
Sub CopyABCDGraph()
```

```
Application.DisplayAlerts = False

    kappa = Sheets("D_Band_Stats").Cells(3, 4)
    r = Sheets("D_Band_Stats").Cells(4, 4)
    p = Sheets("D_Band_Stats").Cells(5, 4)
    r2 = Sheets("D_Band_Stats").Cells(6, 4)

    Windows("AffiliationMacro.xls").Activate
    Sheets("ABCD_Band_Learning_Vs_Cum").Select
    Sheets("ABCD_Band_Learning_Vs_Cum").Copy After:=Workbooks( _
        currFilename).Sheets(Sheets.Count)
    ActiveChart.PlotArea.Select
'   ActiveChart.SeriesCollection(3).Delete
    currChartName = ActiveChart.Name

    timeSteps = CountCols("Affiliation_Cum_Dist_D_Band", 1) - 3

If testBandA > 0 Then

'aband
    j = 14
    While Not (i > 0#)   'if the first time step's mean is zero find the step that doesnt have 0
        i = Sheets("A_Band_Stats").Cells(j, 3).Value
        If Not (i > 0) Then
            j = j + 1
        End If
    Wend

    Charts(currChartName).Select
    'aband mean
    ActiveChart.SeriesCollection(1).XValues = "=A_Band_Stats!R" & j & "C3:R" & timeSteps + 13 & "C3"
    ActiveChart.SeriesCollection(1).Values = "=A_Band_Stats!R" & j & "C6:R" & timeSteps + 13 & "C6"

    'calc y
    ActiveChart.SeriesCollection(2).XValues = "=A_Band_Stats!R" & j + 1 & "C3:R" & timeSteps + 13 &
"C3"
    ActiveChart.SeriesCollection(2).Values = "=A_Band_Stats!R" & j + 1 & "C13:R" & timeSteps + 13 &
"C13"

    '3sigma 3sigma
    ActiveChart.SeriesCollection(3).XValues = "=A_Band_Stats!R" & j + 1 & "C5:R" & timeSteps + 13 &
"C5"
    ActiveChart.SeriesCollection(3).Values = "=A_Band_Stats!R" & j & "C15:R" & timeSteps + 13 &
"C15"

    'aband kappa
    ActiveChart.SeriesCollection(7).XValues = "=A_Band_Stats!R" & j & "C5:R" & timeSteps + 13 & "C5"
    ActiveChart.SeriesCollection(7).Values = "=A_Band_Stats!R" & j & "C10:R" & timeSteps + 13 &
"C10"

    'aband 3sig 3sig
    ActiveChart.SeriesCollection(8).XValues = "=A_Band_Stats!R" & j & "C5:R" & timeSteps + 13 & "C5"
    ActiveChart.SeriesCollection(8).Values = "=A_Band_Stats!R" & timeSteps + 14 & "C8"

    'aband kappa /2
```

```
   ActiveChart.SeriesCollection(9).XValues = "=A_Band_Stats!R" & j & "C5:R" & timeSteps + 13 & "C5"
   ActiveChart.SeriesCollection(9).Values = "=A_Band_Stats!R14C11:R" & timeSteps + 13 & "C11"

   'aband r-p
   ActiveChart.SeriesCollection(10).XValues = "=A_Band_Stats!R" & j & "C12:R" & timeSteps + 13 &
"C12"
   ActiveChart.SeriesCollection(10).Values = "=A_Band_Stats!R14C6:R" & timeSteps + 13 & "C6"

End If 'test band a > 0

   'ActiveChart.SeriesCollection(1).ErrorBars.Select
   'ExecuteExcel4Macro _
   '    "ERRORBAR.Y(2,5,""=A_Band_Stats!R" & j & "C7:R" & timeSteps + 13 &
"C7"",""=A_Band_Stats!$F$" & j & ":$F$" & timeSteps + 13 & """)"

'bband **********THIS IS CORRECT.....
If testBandB > 0 Then
   j = 14
   While Not (i > 0#)   'if the first time step's mean is zero find the step that doesnt have 0
      i = Sheets("B_Band_Stats").Cells(j, 3).Value
      If Not (i > 0) Then
         j = j + 1
      End If
   Wend

'    ActiveChart.SeriesCollection(4).XValues = "=B_Band_Stats!R" & j & "C3:R" & timeSteps + 13 &
"C3"
'    ActiveChart.SeriesCollection(4).Values = "=B_Band_Stats!R" & j & "C6:R" & timeSteps + 13 & "C6"
Else
'    ActiveChart.SeriesCollection(4).XValues = "=0"
'    ActiveChart.SeriesCollection(4).Values = "=0"

End If 'testbandB


'cband
If testBandC > 0 Then
   j = 14
   While Not (i > 0#)   'if the first time step's mean is zero find the step that doesnt have 0
      i = Sheets("C_Band_Stats").Cells(j, 3).Value
      If Not (i > 0) Then
         j = j + 1
      End If
   Wend

   ActiveChart.SeriesCollection(5).XValues = "=C_Band_Stats!R" & j & "C3:R" & timeSteps + 13 & "C3"
   ActiveChart.SeriesCollection(5).Values = "=C_Band_Stats!R" & j & "C6:R" & timeSteps + 13 & "C6"
Else
'    ActiveChart.SeriesCollection(5).XValues = ""
'    ActiveChart.SeriesCollection(5).Values = ""

End If 'testBandC

'dband
If testBandD > 0 Then
   j = 14
```

```
    While Not (i > 0#)   'if the first time step's mean is zero find the step that doesnt have 0
        i = Sheets("D_Band_Stats").Cells(j, 3).Value
        If Not (i > 0) Then
            j = j + 1
        End If
    Wend

'    ActiveChart.SeriesCollection(6).XValues = "=D_Band_Stats!R" & j & "C3:R" & timeSteps + 13 &
"C3"
'    ActiveChart.SeriesCollection(6).Values = "=D_Band_Stats!R" & j & "C6:R" & timeSteps + 13 & "C6"
Else
    ActiveChart.SeriesCollection(6).XValues = "=0"
    ActiveChart.SeriesCollection(6).Values = "=0"

End If 'test band d

    'kappa textbox
'    ActiveChart.Shapes("Text Box 7").Select
'    Selection.Characters.Text = "K= " & kappa & Chr(10) & "r= " & r & Chr(10) & "p= " & p & Chr(10) &
"" & Chr(10) & "R2= " & r2 & Chr(10) & "" & Chr(10) & ""

'    ActiveChart.ChartTitle.Characters.Text = ActiveChart.ChartTitle.Characters.Text & Chr(10) _
'    & technologyName & " (" & stepInterval & ")"

Application.DisplayAlerts = True

End Sub

'--------------------------------------------------------------------------
'  Sub: copyBandSummaryGraphs
'  Author: Matt Behnke
'  Created: 12/13/01
'  Description: Copies the band ENTROPY graphs and published messages summary graphs
'  inputs: band name
'
'  Outputs:
'--------------------------------------------------------------------------

Sub CopyBandSummaryGraphs(ByVal band As String)

Application.DisplayAlerts = False

    If band = "World" Then
        source = "Affiliation_Summary"
    Else
        source = "Affiliation_Summary_" & band
    End If

    numRows = CountRows(source, 1)

'GRAPH ONE message_N_k+1 vs N_k

    Windows("AffiliationMacro.xls").Activate
    Sheets("A_Band_Message_N_k+1 vs N_k").Select
    Sheets("A_Band_Message_N_k+1 vs N_k").Copy Before:=Workbooks(currFilename).Sheets(band &
"_Stats")
```

```
    ActiveChart.SeriesCollection(1).Select

    j = 4
    While Not (i > 0#)   'if the first time step's mean is zero find the step that doesnt have 0
        i = Sheets(source).Cells(j, 3).Value
        If Not (i > 0) Then
            j = j + 1
        End If
    Wend

    If Sheets(source).Cells(i, 2).Characters(1, 1).Text = "1" And Sheets(source).Cells(i, 2).Characters(2,
1).Text = "/" Then
        yx = j
    Else
        yx = j
    End If

    ActiveChart.SeriesCollection(1).XValues = "=" & source & "!R" & j & "C3:R" & numRows - 1 & "C3"
    ActiveChart.SeriesCollection(1).Values = "=" & source & "!R" & j + 1 & "C3:R" & numRows & "C3"
    'y=x:
    ActiveChart.SeriesCollection(2).XValues = "=" & source & "!R" & yx & "C3:R" & numRows & "C3"
    ActiveChart.SeriesCollection(2).Values = "=" & source & "!R" & yx & "C3:R" & numRows & "C3"

    titleBefore = ActiveChart.ChartTitle.Characters.Text

    ActiveChart.ChartTitle.Characters.Text = band & " " & titleBefore & Chr(10) _
    & technologyName & " (" & stepInterval & ")"

    'place subscripts in the chart title (N_k+1, N_k)
    If band = "World" Then
        ActiveChart.ChartTitle.Select
        With Selection.Characters(Start:=29, Length:=3).Font
            .Subscript = True
        End With
        With Selection.Characters(Start:=34, Length:=1).Font
            .Subscript = True
        End With
    Else
        ActiveChart.ChartTitle.Select
        With Selection.Characters(Start:=30, Length:=3).Font
            .Subscript = True
        End With
        With Selection.Characters(Start:=35, Length:=1).Font
            .Subscript = True
        End With
    End If

    ActiveSheet.Name = "" & band & "_Message_N_k+1 vs N_k"

  'copy second graph S_k+1 vs S_k
    Windows("AffiliationMacro.xls").Activate
    Sheets("A_Band_World_S_k+1 vs S_k").Select
    Sheets("A_Band_World_S_k+1 vs S_k").Copy Before:=Workbooks(currFilename).Sheets(band &
"_Stats")
    ActiveChart.SeriesCollection(1).Select
```

```
   If band = "World" Then
      ActiveChart.SeriesCollection(1).XValues = "=" & source & "!R" & j & "C8:R" & numRows - 1 &
"C8"
      ActiveChart.SeriesCollection(1).Values = "=" & source & "!R" & j + 1 & "C8:R" & numRows &
"C8"
      'y=x:
      ActiveChart.SeriesCollection(2).XValues = "=" & source & "!R" & yx & "C8:R" & numRows & "C8"
      ActiveChart.SeriesCollection(2).Values = "=" & source & "!R" & yx & "C8:R" & numRows & "C8"

   Else

      ActiveChart.SeriesCollection(1).XValues = "=" & source & "!R" & j & "C6:R" & numRows - 1 &
"C6"
      ActiveChart.SeriesCollection(1).Values = "=" & source & "!R" & j + 1 & "C6:R" & numRows &
"C6"
      'y=x:
      ActiveChart.SeriesCollection(2).XValues = "=" & source & "!R" & yx & "C6:R" & numRows & "C6"
      ActiveChart.SeriesCollection(2).Values = "=" & source & "!R" & yx & "C6:R" & numRows & "C6"
   End If

   titleBefore = ActiveChart.ChartTitle.Characters.Text

   ActiveChart.ChartTitle.Characters.Text = band & " " & titleBefore & Chr(10) _
   & technologyName & " (" & stepInterval & ")"

   ActiveSheet.Name = "" & band & "_Entropy_S_k+1 vs S_k"

   'place subscripts in the chart title (Entropy S_k+1, S_k)
   If band = "World" Then
      ActiveChart.ChartTitle.Select
      With Selection.Characters(Start:=24, Length:=3).Font
         .Subscript = True
      End With
      With Selection.Characters(Start:=34, Length:=1).Font
         .Subscript = True
      End With
   Else
      ActiveChart.ChartTitle.Select
      With Selection.Characters(Start:=25, Length:=3).Font
         .Subscript = True
      End With
      With Selection.Characters(Start:=35, Length:=1).Font
         .Subscript = True
      End With
   End If


   If band = "World" Then

   Else
'copy third Graph S(Y)_k+1 vs S_world_k
      Windows("AffiliationMacro.xls").Activate
      Sheets("A_Band_S(Y)_k+1 Vs S_world_k").Select
      Sheets("A_Band_S(Y)_k+1 Vs S_world_k").Copy Before:=Workbooks(currFilename).Sheets(band &
"_Stats")
      ActiveChart.SeriesCollection(1).Select
```

```
      ActiveChart.SeriesCollection(1).XValues = "=Affiliation_Summary!R" & j & "C8:R" & numRows &
"C8"
      ActiveChart.SeriesCollection(1).Values = "=" & source & "!R" & j + 1 & "C6:R" & numRows &
"C6"
      'y=x
      ActiveChart.SeriesCollection(2).XValues = "=Affiliation_Summary!R" & yx & "C8:R" & numRows
& "C8"
      ActiveChart.SeriesCollection(2).Values = "=Affiliation_Summary!R" & yx & "C8:R" & numRows &
"C8"

      titleBefore = ActiveChart.ChartTitle.Characters.Text

      ActiveChart.ChartTitle.Characters.Text = band & " " & titleBefore & Chr(10) _
      & technologyName & " (" & stepInterval & ")"

      'subscripts in chart title
      ActiveChart.ChartTitle.Select
         With Selection.Characters(Start:=24, Length:=4).Font
            .Subscript = True
         End With
         With Selection.Characters(Start:=33, Length:=3).Font
            .Subscript = True
         End With
         With Selection.Characters(Start:=39, Length:=5).Font
            .Subscript = True
         End With
         With Selection.Characters(Start:=49, Length:=1).Font
            .Subscript = True
         End With
      ActiveSheet.Name = "" & band & "_S(X,Y)_k+1 vs S_world_k"
   End If

Application.DisplayAlerts = True

End Sub 'copy summary band graphs

'--------------------------------------------------------------------------
'  Sub: FillBandAuthors
'  Author: Matt Behnke
'  Created: 11/7/01
'  Description: fills in a bands author distribution by copying a row from the list of
'               affilations with the number of authors as the matrix's values.
'  inputs: band name
'
'  Outputs:
'--------------------------------------------------------------------------
Sub FillBandAuthors(ByVal band As String)

   Sheets.Add
   sheetName = "Aff_Author_Cum_Dist_" & band & ""
   ActiveSheet.Name = sheetName
   currsheetName = ActiveSheet.Name
   numRowsInBand = CountRows("Affiliation_Cum_Dist_" & band, 1)
   numRowsInAuthors = CountRows("Affiliation_Authors", 1)
```

```
    'Sheets(Worksheets.Count).Select

    Columns("C:C").ColumnWidth = 62.43

    Sheets(currsheetName).Move Before:=Sheets("" & band & "_Stats")

    Sheets("Affiliation_Authors").Select
    Rows("1:1").Select
    Selection.Copy
    Sheets(currsheetName).Select
    Rows("1:1").Select
    ActiveSheet.Paste

    counter = 2
    For i = 2 To numRowsInBand 'copy rows from datasheet into band
       affiliationName = Sheets("Affiliation_Cum_Dist_" & band).Cells(i, 3).Value
       For j = 2 To numRowsInAuthors
          If Sheets("Affiliation_Authors").Cells(j, 3).Value = affiliationName Then
             Sheets("Affiliation_Authors").Select
             Rows(j & ":" & j).Select
             Selection.Copy
             Sheets(currsheetName).Select
             Rows(counter & ":" & counter).Select
             ActiveSheet.Paste
             counter = counter + 1
          End If
       Next j
    Next i

    numRowsInAuthorBand = CountRows(currsheetName, 1)
    numColumns = CountCols(currsheetName, 1) 'num time steps

   Sheets(sheetName).Cells(numRowsInAuthorBand + 1, 3) = "Count"
   Sheets(sheetName).Cells(numRowsInAuthorBand + 2, 3) = "Mean"
   Sheets(sheetName).Cells(numRowsInAuthorBand + 3, 3) = "Std Dev"
   Sheets(sheetName).Cells(numRowsInAuthorBand + 4, 3) = "Sum"

    For i = 4 To numColumns 'put in the mean and std deviation for each time step

'       For j = 2 To numRowsInAuthorBand
'          If (Cells(j, i) > 0 And i > 4) Or (i > 4 Andsheets(sheetName).Cells(j, i - 1) > 0) Then
'             sheets(sheetName).Cells(j, i) =sheets(sheetName).Cells(j, i) +sheets(sheetName).Cells(j, i - 1)
'          End If
'       Next j

       Sheets(sheetName).Cells(numRowsInAuthorBand + 4, i) = "=Sum(" & col(i) & "2:" & col(i) &
numRowsInAuthorBand & ")"
    'add count, avg, stdev...
       Sheets(sheetName).Cells(numRowsInAuthorBand + 1, i).Formula = "=Countif(" & col(i) & "2:" &
col(i) & numRowsInAuthorBand & ", "">0"")"
       If Sheets(sheetName).Cells(numRowsInAuthorBand + 1, i) > 0 Then
          Sheets(sheetName).Cells(numRowsInAuthorBand + 2, i).Formula = "=AVERAGE(" & col(i) & "2:"
& col(i) & numRowsInAuthorBand & ")"
          If Sheets(sheetName).Cells(numRowsInAuthorBand + 1, i) > 1 Then 'more than one so comput std
deviation
```

```
            Sheets(sheetName).Cells(numRowsInAuthorBand + 3, i).Formula = "=STDEV(" & col(i) & "2:"
& col(i) & numRowsInAuthorBand & ")"
          End If
       End If
   Next i


   'Call formatSheetForPrint

End Sub 'band authors
'-------------------------------------------------------------------------
'  Sub: CalcCumulative
'  Author: Matt Behnke
'  Created: 11/15/01
'  Description: processes the input sheet (a matrix) to calculate the cumulative number of
'               instances per time step.
'  inputs: sheetName
'
'  Outputs:
'-------------------------------------------------------------------------
Sub CalcCumulative(ByVal sheetName As String)

   numRows = CountRows(sheetName, 1)
   numCols = CountCols(sheetName, 1)

   Sheets(sheetName).Select

   If sheetName = affiliationDescMatrix Then
      For i = 2 To numRows
         cellSum = 0
         prevSum = 0
         curSum = 0
         For j = 6 To numCols
            prevSum = Sheets(sheetName).Cells(i, j - 1)
            curSum = Sheets(sheetName).Cells(i, j)
            cellSum = prevSum + curSum
            If cellSum > 0 Then
               Sheets(sheetName).Cells(i, j) = cellSum
            End If
         Next j
      Next i
   '
   ' --- For the authors matrix zeros must be put in when
   '     there is no publication in an instance
   '
   ElseIf sheetName = "Affiliation_authors" Or sheetName = dataSheet Then
      For i = 2 To numRows
         cellSum = 0
         prevSum = 0
         curSum = 0
         For j = 4 To numCols
            If j > 4 Then 'when not in first column
               prevSum = Sheets(sheetName).Cells(i, j - 1)
               curSum = Sheets(sheetName).Cells(i, j)
               cellSum = prevSum + curSum
               Sheets(sheetName).Cells(i, j) = cellSum
```

```
            Else 'in first column
               If Not Sheets(sheetName).Cells(i, j) > 0 Then
                  Sheets(sheetName).Cells(i, j) = 0
               End If
            End If

         Next j
      Next i
   Else
      For i = 2 To numRows
         cellSum = 0
         prevSum = 0
         curSum = 0
         For j = 5 To numCols
            prevSum = Sheets(sheetName).Cells(i, j - 1)
            curSum = Sheets(sheetName).Cells(i, j).Value
            If Not curSum = "" And Not prevSum = "" Then
               cellSum = prevSum + curSum
            ElseIf curSum = "" And Not prevSum = "" Then
               cellSum = prevSum
            ElseIf prevSum = "" And Not curSum = "" Then
               cellSum = curSum
            End If
            If cellSum > 0 Then
               Sheets(sheetName).Cells(i, j) = cellSum
            End If
         Next j
      Next i
   End If

   If sheetName = dataSheet Or sheetName = "Affiliation_authors" Then 'put count, mean, and stdev in
each column
      For i = 4 To numCols 'put in the mean and std deviation for each time step
         Sheets(sheetName).Cells(numRows + 4, i).Formula = "=Sum(" & col(i) & "2:" & col(i) &
numRows & ")" 'sum
         Sheets(sheetName).Cells(numRows + 1, i).Formula = "=Countif(" & col(i) & "2:" & col(i) &
numRows & ", "">0"")"
         If Sheets(sheetName).Cells(numRows + 1, i) > 0 Then
            Sheets(sheetName).Cells(numRows + 2, i).Formula = "=AVERAGE(" & col(i) & "2:" & col(i) &
numRows & ")"
            If Sheets(sheetName).Cells(numRows + 1, i) > 1 Then 'more than one so comput std deviation
               Sheets(sheetName).Cells(numRows + 3, i).Formula = "=STDEV(" & col(i) & "2:" & col(i) &
numRows & ")"
            End If
         End If
      Next i
   Else
      'put in the sum of the columns
      For i = 4 To numCols
         Sheets(sheetName).Cells(numRows + 1, i).Formula = "=Sum(" & col(i) & "2:" & col(i) &
numRows & ")"
      Next i
   End If 'sheetname = datasheet
End Sub
```

```
'--------------------------------------------------------------------------
'  Sub: FillBandTerms
'  Author: Matt Behnke
'  Created: 11/7/01
'  Description: fills in the term instances for a band
'  inputs: band name
'
'  Outputs:
'--------------------------------------------------------------------------
Sub FillBandTerms(ByVal band As String)

    sheetName = "Term_Dist_" & band & ""

    Sheets.Add

    ActiveSheet.Name = sheetName
    currsheetName = sheetName

    counter = 2
    affiliationDescMatrix = "descriptor_matrix_affil"
    numRowsInBand = CountRows("Affiliation_Cum_Dist_" & band, 1)
    numColumnsInTerms = CountCols(affiliationDescMatrix, 1)


    Columns("C:C").ColumnWidth = 32.43
    'move the sheet
    Sheets(currsheetName).Move Before:=Sheets("" & band & "_Stats")

    'header
   Sheets(sheetName).Cells(1, 1) = Sheets(affiliationDescMatrix).Cells(1, 1)
   Sheets(sheetName).Cells(1, 2) = Sheets(affiliationDescMatrix).Cells(1, 2)
   Sheets(sheetName).Cells(1, 3) = Sheets(affiliationDescMatrix).Cells(1, 3)

    For i = 4 To numColumnsInTerms - 1 'copy time interval header
      Sheets(sheetName).Cells(1, i) = Sheets("descriptor_matrix_affil").Cells(1, i + 1)
    Next i

    'fill in the terms and instances...
    For i = 2 To numRowsInBand 'copy rows from datasheet into band
       affiliationName = Sheets("Affiliation_Cum_Dist_" & band).Cells(i, 3).Value
       For j = 2 To CountRows(affiliationDescMatrix, 1)
          If Sheets(affiliationDescMatrix).Cells(j, 4) = affiliationName Then

             rowInAffiliationDescMatrix = j
             termName = Sheets(affiliationDescMatrix).Cells(rowInAffiliationDescMatrix, 3)

            'check to see if term exists already on band's list of terms
             termRowInBand = findStringInSheet(currsheetName, termName, 3)

             If termRowInBand > 0 Then

                Sheets(sheetName).Cells(termRowInBand, 2) = Sheets(sheetName).Cells(termRowInBand, 2)
+ Sheets(affiliationDescMatrix).Cells(rowInAffiliationDescMatrix, 2)
                cellSum = 0
                prevSum = 0
                curSum = 0
```

```
            If Sheets(affiliationDescMatrix).Cells(rowInAffiliationDescMatrix, 1).Value <
Sheets(sheetName).Cells(termRowInBand, 1) Then
                 Sheets(sheetName).Cells(termRowInBand, 1) =
Sheets(affiliationDescMatrix).Cells(rowInAffiliationDescMatrix, 1).Value
            End If

            For z = 4 To numColumnsInTerms 'add the values for each time time to what aleady exists
               If z > 4 Then 'add cumulative sum of term instances (previous + current + numInstances)
                  'prevSum = sheets(sheetName).Cells(termRowInBand, z - 1)
                  If Sheets(affiliationDescMatrix).Cells(rowInAffiliationDescMatrix, z + 1) > 0 Then
                     curSum = Sheets(sheetName).Cells(termRowInBand, z) +
Sheets(affiliationDescMatrix).Cells(rowInAffiliationDescMatrix, z + 1)

                  Else
                     curSum = 0
                  End If
                  'cellSum = prevSum + curSum
                  If curSum > 0 Then
                     Sheets(sheetName).Cells(termRowInBand, z) = curSum
                  End If
               Else
                  If Sheets(affiliationDescMatrix).Cells(rowInAffiliationDescMatrix, z + 1) > 0 Then
                     Sheets(sheetName).Cells(termRowInBand, z) =
Sheets(sheetName).Cells(termRowInBand, z) +
Sheets(affiliationDescMatrix).Cells(rowInAffiliationDescMatrix, z + 1)
                  End If
               End If
            Next z
         Else 'term not found
            Sheets(sheetName).Cells(counter, 1) =
Sheets(affiliationDescMatrix).Cells(rowInAffiliationDescMatrix, 1)
            Sheets(sheetName).Cells(counter, 2) =
Sheets(affiliationDescMatrix).Cells(rowInAffiliationDescMatrix, 2)
            Sheets(sheetName).Cells(counter, 3) =
Sheets(affiliationDescMatrix).Cells(rowInAffiliationDescMatrix, 3)
            For z = 4 To numColumnsInTerms
               If z > 4 Then 'add cumulative sum of term instances (previous + current + numInstances)

                  'prevSum = sheets(sheetName).Cells(counter, z - 1)
                  curSum = Sheets(sheetName).Cells(counter, z) +
Sheets(affiliationDescMatrix).Cells(rowInAffiliationDescMatrix, z + 1)
                  'cellSum = prevSum + curSum

                  If curSum > 0 Then
                     Sheets(sheetName).Cells(counter, z) = curSum
               End If
               Else
                  If Sheets(affiliationDescMatrix).Cells(rowInAffiliationDescMatrix, z + 1) > 0 Then
                     Sheets(sheetName).Cells(counter, z) = Sheets(sheetName).Cells(counter, z) +
Sheets(affiliationDescMatrix).Cells(rowInAffiliationDescMatrix, z + 1)
                  End If 'elimates zeros
               End If 'z = 4
            Next z
         counter = counter + 1
         End If 'if-found-else-not
```

```
            End If ' affiliation name matches
        Next j
    Next i

    numRows = CountRows(sheetName, 1)
    numCols = CountCols(sheetName, 1)
        For i = 4 To numCols
            Sheets(sheetName).Cells(numRows + 1, i).Formula = "=Sum(" & col(i) & "2:" & col(i) &
numRows & ")"
        Next i
    'Call CalcCumulative(ActiveSheet.Name)

    Call formatSheetForPrint

End Sub 'fill band terms
'--------------------------------------------------------------------------
'  Sub: FillBandTermsEntropy
'  Author: Matt Behnke
'  Created: 11/17/01
'  Description: computes the entropy of a band's terms.
'               and the contribution of the band..
'  inputs: band name
'
'  Outputs:
'--------------------------------------------------------------------------
Sub FillBandTermsEntropy(ByVal band As String)

    Sheets.Add

    currsheetName = ActiveSheet.Name
    numRows = CountRows("Term_Dist_" & band & "", 1)
    numColumns = CountCols("Term_Dist_" & band & "", 1)

    'Sheets(Worksheets.Count).Select

    Columns("C:C").ColumnWidth = 32.43

    Sheets(currsheetName).Move Before:=Sheets("" & band & "_Stats")

    numRowsWorld = CountRows(descriptorMatrixSheet, 1)

    'copy term distribution sheet for entropy
    Worksheets("Term_Dist_" & band & "").Range("A1:" & col(numColumns) & numRows).Copy
Destination:=Worksheets(currsheetName).Range("A1")


    For i = 2 To numRows
        termName = Sheets(currsheetName).Cells(i, 3)
        termCount = Sheets(currsheetName).Cells(i, 2)
        termRowInWorldEntropy = findStringInSheet(worldEntropySheet, termName, 3)
        termRowInDescriptorMatrix = termRowInWorldEntropy

        For z = 4 To numColumns
            If Sheets(currsheetName).Cells(i, z).Value >= 1 Then
                termCountInBandInStep = Sheets(currsheetName).Cells(i, z)
```

```
            sumInstancesBand = Sheets("Term_Dist_" & band & "").Cells(numRows + 1, z)
            pTerm = termCountInBandInStep / sumInstancesBand
            entropyTerm = -pTerm * (Log(pTerm) / Log(2))

            Sheets(currsheetName).Cells(i, z) = entropyTerm

          End If
        Next z
    Next i

    Sheets(currsheetName).Select
        Sheets(currsheetName).Cells(numRows + 1, 3) = "Sum"
        Sheets(currsheetName).Cells(numRows + 2, 3) = "Contribution"
        Sheets(currsheetName).Cells(numRows + 3, 3) = "Difference"

        For i = 4 To numColumns
          Sheets(currsheetName).Cells(numRows + 1, i).Formula = "=Sum(" & col(i) & "2:" & col(i) &
numRows & ")"

          numInstancesWorld = Sheets(descriptorMatrixSheet).Cells(numRowsWorld + 1, i)
          '3/15/02 - change the num of instances of the world to = the total instances seen at the
          '       last timestep.
          ' this is WRONG... 3/15:


          'set numInstancesWorld = to the total num of instance over all the time steps and terms
          'numInstancesWorld = Sheets(descriptorMatrixSheet).Cells(numRowsWorld + 1, numColumns)


          numInstancesBand = Sheets("Term_Dist_" & band & "").Cells(numRows + 1, i)
          If numInstancesBand > 0 Then
            ratio1 = numInstancesWorld / numInstancesBand
            ratio2 = numInstancesBand / numInstancesWorld

            entropySum = Sheets(currsheetName).Cells(numRows + 1, i)
            contributionOfBand = Abs(ratio2) * entropySum + (ratio2 * (Log(ratio1) / Log(2)))
            Sheets(currsheetName).Cells(numRows + 2, i) = contributionOfBand
            Sheets(currsheetName).Cells(numRows + 3, i) = Abs(entropySum - contributionOfBand)
          Else
            Sheets(currsheetName).Cells(numRows + 2, i) = 0
            Sheets(currsheetName).Cells(numRows + 3, i) = 0
          End If
        Next i

    ActiveSheet.Name = "Term_Entropy_Dist_" & band & ""
'    Call formatSheetForPrint

End Sub 'fill band terms entropy
```

```
'-------------------------------------------------------------------------
'  Sub: affiliationBandSummary
'  Author: Matt Behnke
'  Created: 11/30/01
'  Description: creates the summary sheet for the band..
'               shows step, num of recors, authors, terms, entropy..
'  inputs: band - the name of the band
'  Outputs: none
'-------------------------------------------------------------------------
Sub affiliationBandSummary(ByVal band As String)

    sheetName = "Affiliation_Summary_" & band

    numColumns = CountCols("Term_Entropy_Dist_" & band, 1)
    numRowsAffiliation = CountRows("Affiliation_Cum_Dist_" & band, 1)
    numRowsAuthor = CountRows("Aff_Author_Cum_Dist_" & band, 1)
    numRowsTermDist = CountRows("Term_Dist_" & band, 1)
    numRowsTermEntropy = CountRows("Term_Entropy_Dist_" & band, 1)

    Sheets.Add
    ActiveSheet.Name = sheetName

    currsheetName = ActiveSheet.Name

    Sheets(sheetName).Move Before:=Sheets(band & "_Stats")

    Sheets(currsheetName).Select
    ActiveSheet.StandardWidth = 13

    Sheets(sheetName).Cells(1, 1) = " "
    Sheets(sheetName).Cells(2, 1) = " "

    Sheets(sheetName).Cells(2, 3) = "Instances (Previous + Current)"

    Sheets(sheetName).Cells(3, 1) = "Step"
    Sheets(sheetName).Cells(3, 2) = "interval"

    Sheets(sheetName).Cells(3, 3) = "Records"
    Sheets(sheetName).Cells(3, 4) = "Authors"
    Sheets(sheetName).Cells(3, 5) = "Terms"

    Sheets(sheetName).Cells(3, 6) = "Entropy"
    Sheets(sheetName).Cells(3, 7) = "Contribution"
    Sheets(sheetName).Cells(3, 8) = "Difference"
    Sheets(sheetName).Cells(3, 9) = "Rec / Author"

    For i = 4 To numColumns
        Sheets(sheetName).Cells(i, 1) = i - 3
        Sheets(sheetName).Cells(i, 2) = Sheets("Term_Dist_" & band).Cells(1, i)

        Sheets(sheetName).Cells(i, 3).Value = "=SUM(Affiliation_Cum_Dist_" & band & "!" & col(i) &
"$2:" & col(i) & "$" & numRowsAffiliation & ")"

        Sheets(sheetName).Cells(i, 4).Value = "=SUM(Aff_Author_Cum_Dist_" & band & "!" & col(i) &
"$2:" & col(i) & "$" & numRowsAuthor & ")"
```

```
      Sheets(sheetName).Cells(i, 5).Value = "=SUM(Term_Dist_" & band & "!" & col(i) & "$2:" & col(i)
& "$" & numRowsTermDist & ")"

      Sheets(sheetName).Cells(i, 6) = Sheets("Term_Entropy_Dist_" & band).Cells(numRowsTermEntropy
+ 1, i)
      Sheets(sheetName).Cells(i, 7) = Sheets("Term_Entropy_Dist_" & band).Cells(numRowsTermEntropy
+ 2, i)
      Sheets(sheetName).Cells(i, 8) = Sheets("Term_Entropy_Dist_" & band).Cells(numRowsTermEntropy
+ 3, i)
      If Sheets(sheetName).Cells(i, 4) > 0 Then
         Sheets(sheetName).Cells(i, 9) = Sheets(sheetName).Cells(i, 3) / Sheets(sheetName).Cells(i, 4)
      End If
   Next i


End Sub

'--------------------------------------------------------------------------
'  Sub: affiliationSummary
'  Author: Matt Behnke
'  Created: 11/30/01
'  added stuff: 2/1/02
'  Description: creates the world affilation summary sheet..
'               this is the first part.. the second part puts in the temp poly and the pressure equations
'               after fill months has been run on the sheet.....
'  inputs: none
'  Outputs: none
'--------------------------------------------------------------------------
Sub affiliationSummary()

   numColumns = CountCols(worldEntropySheet, 1)
   numRowsAffiliation = CountRows(dataSheet, 1)
   numRowsAuthor = CountRows("Affiliation_authors", 1)
   numRowsTermDist = CountRows(descriptorMatrixSheet, 1)
   numRowsTermEntropy = CountRows(worldEntropySheet, 1)

   Sheets.Add After:=Worksheets(Worksheets.Count)
'   Sheets(Worksheets.Count).Select
   sheetName = "Affiliation_Summary"
   ActiveSheet.Name = sheetName
   currsheetName = ActiveSheet.Name

   Sheets(currsheetName).Move After:=Sheets(Sheets.Count)

   Sheets(currsheetName).Select
   ActiveSheet.StandardWidth = 13


   Sheets(sheetName).Cells(1, 1) = " "
   Sheets(sheetName).Cells(2, 1) = " "

   Sheets(sheetName).Cells(2, 3) = "Instances (Previous + Current)"

   Sheets(sheetName).Cells(3, 1) = "Step"
   Sheets(sheetName).Cells(3, 2) = "interval"
```

```
    Sheets(sheetName).Cells(3, 3) = "Records"
    Sheets(sheetName).Cells(3, 4) = "Authors (v_X)"
    Sheets(sheetName).Cells(3, 5) = "Rec / Author"
    Sheets(sheetName).Cells(3, 6) = "Terms X"
    Sheets(sheetName).Cells(3, 7) = "Terms Y"
    Sheets(sheetName).Cells(3, 8) = "S(X)"
    Sheets(sheetName).Cells(3, 9) = "S(Y)"
    Sheets(sheetName).Cells(3, 10) = "S(X,Y)"
    Sheets(sheetName).Cells(3, 11) = "S(X;Y)"
    Sheets(sheetName).Cells(3, 12) = "delta_n_x"
    Sheets(sheetName).Cells(3, 13) = "delta_s_x"
    Sheets(sheetName).Cells(3, 14) = "T_X Saboe Degrees"
    Sheets(sheetName).Cells(3, 15) = "delta_n_y"
    Sheets(sheetName).Cells(3, 16) = "delta_s_y"
    Sheets(sheetName).Cells(3, 17) = "v_Y_nodes"
    Sheets(sheetName).Cells(3, 18) = "pressure_n per node"

    For i = 4 To numColumns
        Sheets(sheetName).Cells(i, 1) = i - 3
        Sheets(sheetName).Cells(i, 2) = Sheets(dataSheet).Cells(1, i)

        If i > 4 Then
            Sheets(sheetName).Cells(i, 3).Value = "=SUM('" & dataSheet & "'!" & col(i) & "$2:" & col(i) &
"$" & numRowsAffiliation & ")" '+ C" & i - 1
        Else
            Sheets(sheetName).Cells(i, 3).Value = "=SUM('" & dataSheet & "'!" & col(i) & "$2:" & col(i) &
"$" & numRowsAffiliation & ")"
        End If

        If i > 4 Then
            Sheets(sheetName).Cells(i, 4).Value = "=SUM(Affiliation_authors!" & col(i) & "$2:" & col(i) &
"$" & numRowsAuthor & ")" ' + D" & i - 1
        Else
            Sheets(sheetName).Cells(i, 4).Value = "=SUM(Affiliation_authors!" & col(i) & "$2:" & col(i) &
"$" & numRowsAuthor & ")"
        End If

        Sheets(sheetName).Cells(i, 5) = Sheets(sheetName).Cells(i, 3) / Sheets(sheetName).Cells(i, 4)
        Sheets(sheetName).Cells(i, 6).Value = "=SUM(" & descriptorMatrixSheet & "!" & col(i) & "$2:" &
col(i) & "$" & numRowsTermDist & ")"
        Sheets(sheetName).Cells(i, 7).Value = "=SUM(" & descriptorMatrixSheetY & "!" & col(i) & "$2:" &
col(i) & "$" & numRowsTermDist & ")"
        Sheets(sheetName).Cells(i, 8) = Sheets(worldEntropySheet).Cells(numRowsTermEntropy + 1, i)
        Sheets(sheetName).Cells(i, 9) = Sheets(worldEntropySheetY).Cells(numRowsTermEntropy + 1, i)
        Sheets(sheetName).Cells(i, 10) = Sheets(worldEntropySheet).Cells(numRowsTermEntropy + 1,
numColumns)
        Sheets(sheetName).Cells(i, 11) = "=" & col(8) & i & "+" & col(9) & i & "-" & col(10) & i 'Cells(i, 8)
+ sheets(sheetName).Cells(i, 9) - sheets(sheetName).Cells(i, 10)

        If i > 4 Then
            Sheets(sheetName).Cells(i, 12) = "=" & col(6) & i & "-" & col(6) & i - 1 'cells(i, 6) - cells (i-1,6)
delta_n_y
            Sheets(sheetName).Cells(i, 13) = "=" & col(9) & i - 1 & "-" & col(9) & i 'cells(i, 9) - cells (i-1,9)
delta_s_x
            Sheets(sheetName).Cells(i, 14) = "=" & col(12) & i & "/" & col(13) & i 'cells(i, 12) / cells(i, 13)
T_X
```

```
        Sheets(sheetName).Cells(i, 15) = "=" & col(7) & i & "-" & col(7) & i - 1 'cells(i, 7) - cells (i-1,7)
delta_n_y
        Sheets(sheetName).Cells(i, 16) = "=" & col(8) & i & "-" & col(8) & i - 1 'cells(i, 8) - cells(i-1, 8)
delta_s_y
      End If

      Sheets(sheetName).Cells(i, 17) = Sheets("Affiliation_authors").Cells(numRowsAuthor + 1,
numColumns) - Sheets(sheetName).Cells(i, 4)
      Sheets(sheetName).Cells(i, 18) = "=" & col(6) & i & "/" & col(4) & i 'cells(i,6) / cells(i,4) terms X /
author X
   Next i

   Sheets(sheetName).Cells(4, 3).Select 'freeze panes
   ActiveWindow.FreezePanes = True


   Call fillMonthsRow("Affiliation_Summary", 4)
'   Call fillMonthsRow("Affiliation_Summary", 4)
   'Call CopyInteractingSystemsGraphs(ActiveSheet.Name, numColumns)
   numRows = CountRows("Affiliation_Summary", 1)
   Call FillInMissingData("Affiliation_Summary", 6, 4, numRows)
   Call FillInMissingData("Affiliation_Summary", 9, 4, numRows)

End Sub

'--------------------------------------------------------------------------
'  Sub: affiliationSummaryPart2
'  Author: Matt Behnke
'  Created: 2/1/02
'  Description: after fillmonths has been ran this procedure copies the appropriate graphs
'               interactive systems graphs and temp / pressure graphs
'               uses the trendline equations from the system graph to calculate
'               temp_polynomial and
'               the pressure equation
'  inputs: none
'  Outputs: none
'--------------------------------------------------------------------------
Sub affiliationSummaryPart2()

   source = "Affiliation_Summary"
   numRows = CountRows(source, 1)

   Sheets(source).Cells(3, 19) = "S(X) calculated"
   Sheets(source).Cells(3, 20) = "S(Y) calculated"
   Sheets(source).Cells(3, 21) = "delta S(X) calculated"
   Sheets(source).Cells(3, 22) = "n(X) calculated"
   Sheets(source).Cells(3, 23) = "delta_n_x_calculated"
   Sheets(source).Cells(3, 24) = "T_X Saboe Deg. Polynomial"

   Call CopyInteractingSystemsGraphs("World")

   trendlineA = Sheets(source).Cells(1, 19)
   sx_a = firstPartTrendEq(trendlineA)
   sx_b = secondPartTrendEq(trendlineA)
   'sx_a = firstPartPolyTrendEq(trendlineA)
   'sx_b = secondPartPolyTrendEq(trendlineA)
```

```
'sx_c = thirdPartPolyTrendEq(trendlineA)

trendlineB = Sheets(source).Cells(1, 20)
sy_a = firstPartPolyTrendEq(trendlineB)
sy_b = secondPartPolyTrendEq(trendlineB)
sy_c = thirdPartPolyTrendEq(trendlineB)

trendline_nX = Sheets(source).Cells(1, 22)
nx_a = firstPartTrendEq(trendline_nX)
nx_b = secondPartTrendEq(trendline_nX)

For i = 4 To numRows
   k = Sheets(source).Cells(i, 1)
   'sX & sY calculated
   Sheets(source).Cells(i, 19) = sx_a * k ^ sx_b 'power equation of entropy
   'Sheets(source).Cells(i, 19) = sx_a * k ^ 2 + sx_b * k + sx_c
   Sheets(source).Cells(i, 20) = sy_a * k ^ 2 + sy_b * k + sy_c
   'nX calculated
   Sheets(source).Cells(i, 22) = nx_a * (k ^ nx_b)


   If Sheets(source).Cells(i - 1, 6).Font.ColorIndex = 3 Then
      'find the first row of the same value
      x = i - 1
      While Sheets(source).Cells(x, 6).Font.ColorIndex = 3
         x = x - 1
      Wend
      previous_SY = Sheets(source).Cells(x, 9) 'S(Y) from previous step
      previous_nX = Sheets(source).Cells(x, 6) 'number of terms in previous step
   Else
      previous_SY = Sheets(source).Cells(i - 1, 9) 'S(Y) from previous step
      previous_nX = Sheets(source).Cells(i - 1, 6) 'number of terms in previous step
   End If

   If i > 4 Then
      'check to see if current S(Y) or current n(X) (num terms) is the same as previous
      'if so then place the value of the calculated S(Y) or n(X) into that spot of similarity
      'mark the spot in red where a calculated value has been substituted.
      If Sheets(source).Cells(i, 9) = previous_SY Then
         Sheets(source).Cells(i, 9) = "=" & col(20) & i 'equals calc'ed value of S(Y)
         Sheets(source).Cells(i, 9).Font.ColorIndex = 3
      End If
      If Sheets(source).Cells(i, 6) = previous_nX Then
         Sheets(source).Cells(i, 6) = "=" & col(22) & i 'equals calc'ed value of n(X)
         Sheets(source).Cells(i, 6).Font.ColorIndex = 3
      End If

      'delta S(X)_calculated
      Sheets(source).Cells(i, 21) = "=" & col(19) & i & "-" & col(19) & i - 1 'cells(i,19) - cells(i-1,19)
      'delta n(X)_calculated
      Sheets(source).Cells(i, 23) = "=" & col(22) & i & "-" & col(22) & i - 1 'cells(i,22) - cells(i-1,22)
      't(x)_poly = n(X)/S(X)
      Sheets(source).Cells(i, 24) = "=" & col(23) & i & "/" & col(21) & i 'cells(i,23) / cells(i,21)
   End If
Next i
```

```
End Sub

'--------------------------------------------------------------------------
'  Sub: affiliationSummaryPart3
'  Author: Matt Behnke
'  Created: 2/4/02
'  Description: copies the temp / pressure graphs uses trendline equations of temp_poly and pressure to get
the
'               the pressure equation
'  inputs: none
'  Outputs: none
'--------------------------------------------------------------------------
Sub affiliationSummaryPart3()

    source = "Affiliation_Summary"
    numRows = CountRows(source, 1)

    Sheets(source).Cells(3, 25) = "Press f(T)"
    Sheets(source).Cells(1, 24) = "m_P"
    Sheets(source).Cells(2, 24) = "b_P"
    Sheets(source).Cells(1, 26) = "m_T"
    Sheets(source).Cells(2, 26) = "b_T"

    Call CopyTempPressGraphs("World")

    trendline_Tpoly = Sheets(source).Cells(1, 27)
    m_t = firstPartTrendEq(trendline_Tpoly)
    b_t = secondPartLinearTrendEq(trendline_Tpoly)
    Sheets(source).Cells(1, 27) = m_t
    Sheets(source).Cells(2, 27) = b_t

    trendline_Press = Sheets(source).Cells(1, 25)
    m_p = firstPartTrendEq(trendline_Press)
    b_p = secondPartLinearTrendEq(trendline_Press)
    Sheets(source).Cells(1, 25) = m_p
    Sheets(source).Cells(2, 25) = b_p


    For i = 5 To numRows
        Tx_poly = Sheets(source).Cells(i, 24)
        Sheets(source).Cells(i, 25) = b_p + (m_p / m_t) * (Tx_poly - b_t)
    Next i

'copy third Graph World_Press_vs_Temp_Saboe
    Application.DisplayAlerts = False

    Windows("AffiliationMacro.xls").Activate
    Sheets("World_Press_vs_Temp_Saboe").Select
    Sheets("World_Press_vs_Temp_Saboe").Copy After:=Workbooks(currFilename).Sheets(source)
    ActiveChart.SeriesCollection(1).Select

    'Pressure per node
    ActiveChart.SeriesCollection(1).Values = "=" & source & "!R" & 5 & "C25:R" & numRows & "C25"
    ActiveChart.SeriesCollection(1).XValues = "=" & source & "!R" & 5 & "C24:R" & numRows & "C24"
    'T(x) poly:
    ActiveChart.SeriesCollection(2).Values = "=" & source & "!R" & 5 & "C18:R" & numRows & "C18"
```

```
    ActiveChart.SeriesCollection(2).XValues = "=" & source & "!R" & 5 & "C24:R" & numRows & "C24"

    Application.DisplayAlerts = True

End Sub

'--------------------------------------------------------------------------
'  Sub: copyTempPressGraphs
'  Author: Matt Behnke
'  Created: 2/4/02
'  Description: copies the interacting systems graphs from the affiliation macro workbook.
'  inputs: band name
'
'  Outputs:
'--------------------------------------------------------------------------
Sub CopyTempPressGraphs(ByVal band As String)

    Application.DisplayAlerts = False

    If band = "World" Then
        source = "Affiliation_Summary"
    Else
        source = "Affiliation_Summary_" & band
    End If

    numRows = CountRows(source, 1)

'GRAPH ONE XY_Temp

    Windows("AffiliationMacro.xls").Activate
    Sheets("XY_Temp").Select
    Sheets("XY_Temp").Copy After:=Workbooks(currFilename).Sheets(source)
    ActiveChart.SeriesCollection(1).Select

    j = 4
    While Not (i > 0#)   'if the first time step's mean is zero find the step that doesnt have 0
        i = Sheets(source).Cells(j, 3).Value
        If Not (i > 0) Then
            j = j + 1
        End If
    Wend

    'X-Category

    'msgs per node
    ActiveChart.SeriesCollection(1).Values = "=" & source & "!R" & j & "C18:R" & numRows & "C18"
    ActiveChart.SeriesCollection(1).XValues = "=" & source & "!R" & j & "C1:R" & numRows & "C1"
    't(x)
    ActiveChart.SeriesCollection(2).Values = "=" & source & "!R" & j & "C14:R" & numRows & "C14"
    ActiveChart.SeriesCollection(2).XValues = "=" & source & "!R" & j & "C1:R" & numRows & "C1"
    't(x) poly
    ActiveChart.SeriesCollection(3).Values = "=" & source & "!R" & j & "C24:R" & numRows & "C24"
    ActiveChart.SeriesCollection(3).XValues = "=" & source & "!R" & j & "C18:R" & numRows & "C18"

    With ActiveChart.SeriesCollection(2).Trendlines(1)
    'put trendline equation onto stats sheet for T(X)_poly
```

```
    .DisplayEquation = True
    .DisplayRSquared = True

   End With

    With ActiveChart.SeriesCollection(3).Trendlines(1)
    'put trendline equation onto stats sheet for T(X)_poly
    .DisplayEquation = True
    .DisplayRSquared = True
     Worksheets(source).Cells(1, 27).Value = .DataLabel.Text
   End With

  'copy second graph XY_Press
    Windows("AffiliationMacro.xls").Activate
    Sheets("XY_Press").Select
    Sheets("XY_Press").Copy After:=Workbooks(currFilename).Sheets(source)

    ActiveChart.SeriesCollection(1).Select

    'Pressure per node
    ActiveChart.SeriesCollection(1).Values = "=" & source & "!R" & j & "C18:R" & numRows & "C18"
    ActiveChart.SeriesCollection(1).XValues = "=" & source & "!R" & j & "C1:R" & numRows & "C1"
    'T(x) poly:
    ActiveChart.SeriesCollection(2).Values = "=" & source & "!R" & j & "C24:R" & numRows & "C24"
    ActiveChart.SeriesCollection(2).XValues = "=" & source & "!R" & j & "C18:R" & numRows & "C18"

    With ActiveChart.SeriesCollection(2).Trendlines(1)
    'put trendline equation onto stats sheet for pressure fit
    .DisplayEquation = True
    .DisplayRSquared = True
    End With

    With ActiveChart.SeriesCollection(1).Trendlines(1)
    'put trendline equation onto stats sheet for pressure fit
    .DisplayEquation = True
    .DisplayRSquared = True
     Worksheets(source).Cells(1, 25).Value = .DataLabel.Text
    End With

    Application.DisplayAlerts = True

End Sub 'copy temp/press graphs


'---------------------------------------------------------------------------
'  Sub: copyInteractingSystemsGraphs
'  Author: Matt Behnke
'  Created: 2/1/02
'  Description: copies the interacting systems graphs from the affiliation macro workbook.
'  inputs: band name
'
'  Outputs:
'---------------------------------------------------------------------------
Sub CopyInteractingSystemsGraphs(ByVal band As String)

    Application.DisplayAlerts = False
```

```
    If band = "World" Then
       source = "Affiliation_Summary"
    Else
       source = "Affiliation_Summary_" & band
    End If

    numRows = CountRows(source, 1)

'GRAPH ONE S_2Interacting Systems

    Windows("AffiliationMacro.xls").Activate
    Sheets("S_2Interacting systems").Select
    Sheets("S_2Interacting systems").Copy After:=Workbooks(currFilename).Sheets(source)
    ActiveChart.SeriesCollection(1).Select

    j = 4
    While Not (i > 0#)   'if the first time step's mean is zero find the step that doesnt have 0
       i = Sheets(source).Cells(j, 3).Value
       If Not (i > 0) Then
          j = j + 1
       End If
    Wend

    'X-Category

    'S(Y)
    ActiveChart.SeriesCollection(1).Values = "=" & source & "!R" & j & "C9:R" & numRows & "C9"
    ActiveChart.SeriesCollection(1).XValues = "=" & source & "!R" & j & "C6:R" & numRows & "C6"
    'S(X):
    ActiveChart.SeriesCollection(2).Values = "=" & source & "!R" & j & "C8:R" & numRows & "C8"
    ActiveChart.SeriesCollection(2).XValues = "=" & source & "!R" & j & "C6:R" & numRows & "C6"
    'S(X,Y)
    ActiveChart.SeriesCollection(3).Values = "=" & source & "!R" & j & "C10:R" & numRows & "C10"
    ActiveChart.SeriesCollection(3).XValues = "=" & source & "!R" & j & "C6:R" & numRows & "C6"
    'S(X;Y)
    ActiveChart.SeriesCollection(4).Values = "=" & source & "!R" & j & "C11:R" & numRows & "C11"
    ActiveChart.SeriesCollection(4).XValues = "=" & source & "!R" & j & "C6:R" & numRows & "C6"


    With ActiveChart.SeriesCollection(1).Trendlines(1)
    'put trendline equation onto stats sheet for S(y)
    .DisplayEquation = True
    .DisplayRSquared = True
     Worksheets(source).Cells(1, 20).Value = .DataLabel.Text
   End With

   With ActiveChart.SeriesCollection(2).Trendlines(1)
    'put trendline equation onto stats sheet for S(x)
    .DisplayEquation = True
    .DisplayRSquared = True
     Worksheets(source).Cells(1, 19).Value = .DataLabel.Text
   End With

  'copy second graph World_(X)Temp_S_2
    Windows("AffiliationMacro.xls").Activate
```

```
    Sheets("World_(X)Temp_S_2").Select
    Sheets("World_(X)Temp_S_2").Copy After:=Workbooks(currFilename).Sheets(source)

    ActiveChart.SeriesCollection(1).Select

    'S(X) vs T_X
    ActiveChart.SeriesCollection(1).Values = "=" & source & "!R" & j & "C8:R" & numRows & "C8"
    ActiveChart.SeriesCollection(1).XValues = "=" & source & "!R" & j & "C14:R" & numRows & "C14"
    'S(X;Y):
    ActiveChart.SeriesCollection(2).Values = "=" & source & "!R" & j & "C11:R" & numRows & "C11"
    ActiveChart.SeriesCollection(2).XValues = "=" & source & "!R" & j & "C6:R" & numRows & "C6"

    With ActiveChart.SeriesCollection(1).Trendlines(1)
    'put trendline equation onto stats sheet for S(y)
    .DisplayEquation = True
    .DisplayRSquared = True

   End With

   With ActiveChart.SeriesCollection(2).Trendlines(1)
    'put trendline equation onto stats sheet for S(x)
    .DisplayEquation = True
    .DisplayRSquared = True

   End With

'copy third Graph n_Msg_2Interacting systems
    Windows("AffiliationMacro.xls").Activate
    Sheets("n_Msg_2Interacting systems").Select
    Sheets("n_Msg_2Interacting systems").Copy After:=Workbooks(currFilename).Sheets(source)
    ActiveChart.SeriesCollection(1).Select

    'n_X
    ActiveChart.SeriesCollection(1).Values = "=" & source & "!R" & j & "C6:R" & numRows & "C6"

    With ActiveChart.SeriesCollection(1).Trendlines(1)
    'put trendline equation onto stats sheet for S(x)
    .DisplayEquation = True
    .DisplayRSquared = True
     Worksheets(source).Cells(1, 22).Value = .DataLabel.Text
    End With

    'n_Y
    ActiveChart.SeriesCollection(2).Values = "=Affiliation_Summary!R" & j & "C7:R" & numRows &
"C7"

    Application.DisplayAlerts = True

End Sub 'copy interacting systems graphs

'--------------------------------------------------------------------------
'  Sub: entropySummary
'  Author: Matt Behnke
'  Created: 11/19/01
'  Description: creates the entropy summary sheet.. for the world and all the bands,
'               shows the local and contribution entropies of each band
```

```
'  inputs: none
'  Outputs: none
'-------------------------------------------------------------------------
Sub entropySummary()

    sheetName = "Entropy Summary"
    numColumnsInTerms = CountCols(worldEntropySheet, 1)
If testBandA > 0 Then
    numRowsAband = CountRows("Term_Entropy_Dist_A_Band", 1)
End If

If testBandB > 0 Then
    numRowsBband = CountRows("Term_Entropy_Dist_B_Band", 1)
End If

If testBandC > 0 Then
    numRowsCband = CountRows("Term_Entropy_Dist_C_Band", 1)
End If

If testBandD > 0 Then
    numRowsDband = CountRows("Term_Entropy_Dist_D_Band", 1)
End If

    numRowsWorld = CountRows(worldEntropySheet, 1)

    Sheets.Add After:=Worksheets(Worksheets.Count)
'    Sheets(Worksheets.Count).Select
    ActiveSheet.Name = sheetName
    currsheetName = sheetName

    Sheets(currsheetName).Move After:=Sheets("D_Band_Stats")

    Sheets(currsheetName).Select
    ActiveSheet.StandardWidth = 13

    Sheets(sheetName).Cells(1, 1) = " "
    Sheets(sheetName).Cells(2, 1) = " "

    Sheets(sheetName).Cells(3, 1) = "Step"
    Sheets(sheetName).Cells(3, 2) = "interval"

    Sheets(sheetName).Cells(3, 3) = "A_Band Entropy"
    Sheets(sheetName).Cells(3, 4) = "A_Band Contribution"
    Sheets(sheetName).Cells(3, 5) = "A_Band Difference"

    Sheets(sheetName).Cells(3, 6) = "B_Band Entropy"
    Sheets(sheetName).Cells(3, 7) = "B_Band Contribution"
    Sheets(sheetName).Cells(3, 8) = "B_Band Difference"

    Sheets(sheetName).Cells(3, 9) = "C_Band Entropy"
    Sheets(sheetName).Cells(3, 10) = "C_Band Contribution"
    Sheets(sheetName).Cells(3, 11) = "C_Band Difference"

    Sheets(sheetName).Cells(3, 12) = "D_Band Entropy"
    Sheets(sheetName).Cells(3, 13) = "D_Band Contribution"
    Sheets(sheetName).Cells(3, 14) = "D_Band Difference"
```

```
    Sheets(sheetName).Cells(3, 15) = "Sum Band Entropy"
    Sheets(sheetName).Cells(3, 16) = "Sum Band Contribution"
    Sheets(sheetName).Cells(3, 17) = "World Entropy"
    Sheets(sheetName).Cells(3, 18) = "Diff World & Contrib"


    For i = 4 To numColumnsInTerms
        Sheets(sheetName).Cells(i, 1) = i - 3
        Sheets(sheetName).Cells(i, 2) = Sheets("Term_Entropy_Dist_D_Band").Cells(1, i)
        If testBandA > 0 Then


            Sheets(sheetName).Cells(i, 3) = Sheets("Term_Entropy_Dist_A_Band").Cells(numRowsAband + 1,
i)
            Sheets(sheetName).Cells(i, 4) = Sheets("Term_Entropy_Dist_A_Band").Cells(numRowsAband + 2,
i)
            Sheets(sheetName).Cells(i, 5) = Sheets("Term_Entropy_Dist_A_Band").Cells(numRowsAband + 3,
i)
        End If

        If testBandB > 0 Then
            Sheets(sheetName).Cells(i, 6) = Sheets("Term_Entropy_Dist_B_Band").Cells(numRowsBband + 1,
i)
            Sheets(sheetName).Cells(i, 7) = Sheets("Term_Entropy_Dist_B_Band").Cells(numRowsBband + 2,
i)
            Sheets(sheetName).Cells(i, 8) = Sheets("Term_Entropy_Dist_B_Band").Cells(numRowsBband + 3,
i)
        End If

        If testBandC > 0 Then
            Sheets(sheetName).Cells(i, 9) = Sheets("Term_Entropy_Dist_C_Band").Cells(numRowsCband + 1,
i)
            Sheets(sheetName).Cells(i, 10) = Sheets("Term_Entropy_Dist_C_Band").Cells(numRowsCband +
2, i)
            Sheets(sheetName).Cells(i, 11) = Sheets("Term_Entropy_Dist_C_Band").Cells(numRowsCband +
3, i)
        End If

        If testBandD > 0 Then
            Sheets(sheetName).Cells(i, 12) = Sheets("Term_Entropy_Dist_D_Band").Cells(numRowsDband +
1, i)
            Sheets(sheetName).Cells(i, 13) = Sheets("Term_Entropy_Dist_D_Band").Cells(numRowsDband +
2, i)
            Sheets(sheetName).Cells(i, 14) = Sheets("Term_Entropy_Dist_D_Band").Cells(numRowsDband +
3, i)
        End If

        Sheets(sheetName).Cells(i, 15) = Sheets(sheetName).Cells(i, 3) + Sheets(sheetName).Cells(i, 6) +
Sheets(sheetName).Cells(i, 9) + Sheets(sheetName).Cells(i, 12)
        Sheets(sheetName).Cells(i, 16) = Sheets(sheetName).Cells(i, 4) + Sheets(sheetName).Cells(i, 7) +
Sheets(sheetName).Cells(i, 10) + Sheets(sheetName).Cells(i, 13)
        Sheets(sheetName).Cells(i, 17) = Sheets(worldEntropySheet).Cells(numRowsWorld + 1, i)
        Sheets(sheetName).Cells(i, 18) = Abs(Cells(i, 17) - Sheets(sheetName).Cells(i, 16))
    Next i
```

```
End Sub

'--------------------------------------------------------------------------
'  Sub: instancesSummary
'  Author: Matt Behnke
'  Created: 4/10/02
'  Description: creates a summary sheet of the number of instances in each band over time
'               this is used to compare the number of instances in each band to the instances
'               for the world.
'               IF they are different then the source data for the band instances is not consistent
'               with the source data for the world instances. (descriptor_matrix_affil vs. descriptor data_x)
'  inputs: none
'  Outputs: none
'--------------------------------------------------------------------------
Sub instancesSummary()

'testBandA = 1
'testBandB = 1
'testBandC = 1
'testBandD = 1

    sheetName = "Instances Summary"
    numColumnsInTerms = CountCols(descriptorMatrixSheet, 1)
If testBandA > 0 Then
    numRowsAband = CountRows("Term_Dist_A_Band", 1)
End If

If testBandB > 0 Then
    numRowsBband = CountRows("Term_Dist_B_Band", 1)
End If

If testBandC > 0 Then
    numRowsCband = CountRows("Term_Dist_C_Band", 1)
End If

If testBandD > 0 Then
    numRowsDband = CountRows("Term_Dist_D_Band", 1)
End If

    numRowsWorld = CountRows(descriptorMatrixSheet, 1)

    Sheets.Add After:=Worksheets(Worksheets.Count)
'    Sheets(Worksheets.Count).Select
    ActiveSheet.Name = sheetName
    currsheetName = sheetName

    Sheets(currsheetName).Move After:=Sheets(descriptorMatrixSheet)

    Sheets(currsheetName).Select
    ActiveSheet.StandardWidth = 13

    Sheets(sheetName).Cells(1, 1) = " "
    Sheets(sheetName).Cells(2, 1) = " "

    Sheets(sheetName).Cells(3, 1) = "Step"
```

```
    Sheets(sheetName).Cells(3, 2) = "interval"

    Sheets(sheetName).Cells(3, 3) = "A_Band Instances"
    Sheets(sheetName).Cells(3, 4) = "B_Band Instances"
    Sheets(sheetName).Cells(3, 5) = "C_Band Instances"
    Sheets(sheetName).Cells(3, 6) = "D_Band Entropy"

    Sheets(sheetName).Cells(3, 7) = "Sum Band Instances"

    Sheets(sheetName).Cells(3, 8) = "World Entropy Instances"
    Sheets(sheetName).Cells(3, 9) = "Diff World & Band"


    For i = 4 To numColumnsInTerms
        Sheets(sheetName).Cells(i, 1) = i - 3
        Sheets(sheetName).Cells(i, 2) = Sheets("Term_Entropy_Dist_D_Band").Cells(1, i)
        If testBandA > 0 Then

            Sheets(sheetName).Cells(i, 3) = Sheets("Term_Dist_A_Band").Cells(numRowsAband + 1, i)

        End If

        If testBandB > 0 Then
            Sheets(sheetName).Cells(i, 4) = Sheets("Term_Dist_B_Band").Cells(numRowsBband + 1, i)
        End If

        If testBandC > 0 Then
            Sheets(sheetName).Cells(i, 5) = Sheets("Term_Dist_C_Band").Cells(numRowsCband + 1, i)
        End If

        If testBandD > 0 Then
            Sheets(sheetName).Cells(i, 6) = Sheets("Term_Dist_D_Band").Cells(numRowsDband + 1, i)
        End If

        Sheets(sheetName).Cells(i, 7) = Sheets(sheetName).Cells(i, 3) + Sheets(sheetName).Cells(i, 4) +
Sheets(sheetName).Cells(i, 5) + Sheets(sheetName).Cells(i, 6)

        Sheets(sheetName).Cells(i, 8) = Sheets(descriptorMatrixSheet).Cells(numRowsWorld + 1, i)
        Sheets(sheetName).Cells(i, 9) = Abs(Cells(i, 8) - Sheets(sheetName).Cells(i, 7))
    Next i


'testBandA = 0
'testBandB = 0
'testBandC = 0
'testBandD = 0

End Sub


'---------------------------------------------------------------------
'  Function: findStringRowInSheet
'  Author: Matt Behnke
'  Created: 2/28/02
'  Description: determines the row of the string in the given sheet. uses find function
'  inputs: matrixSheet, termName (descriptor), column letter of term in matrixSheet
```

```
'  Outputs: the row number
'-------------------------------------------------------------------------
Function findStringInSheet(ByVal matrixSheet As String, ByVal termName As String, ByVal column As
String) As String

With Worksheets(matrixSheet).Range(column & ":" & column)
   Set C = .Find(termName, LookIn:=xlValues)
   If Not C Is Nothing Then
      firstAddress = C.Address
      temp = Sheets(1).Cells(1, 1)
      Sheets(1).Cells(1, 1) = firstAddress
      theRow = Sheets(1).Cells(1, 1).Characters(4, 5).Text
      Sheets(1).Cells(1, 1) = temp



      findStringInSheet = theRow
   Else
      findStringInSheet = 0
   End If
End With




End Function 'funciton

'-------------------------------------------------------------------------
'  Function: findStringRowInSheet ****OBSOLETE*** Slow
'  Author: Matt Behnke
'  Created: 11/16/01
'  Description: determines the row of the string in the given sheet
'  inputs: sheetname, descriptor, column of desc in datasheet
'  Outputs: row number where the value is found
'-------------------------------------------------------------------------
Function findStringRowInSheet(ByVal matrixSheet As String, ByVal termName As String, ByVal
columnNum As Integer) As Integer

   foundAt = 0
   numRows = CountRows(matrixSheet, 1)
   For i = 2 To numRows 'assume column header
      If Cells(i, columnNum).Value = termName Then
         foundAt = i
         found = True
         Exit For
      End If
   Next i
   If found = True Then
      findStringRowInSheet = foundAt
   Else
      findStringRowInSheet = 0
   End If

End Function
'-------------------------------------------------------------------------
'  Subroutine: formatSheetForPrint
```

```
'  Author: Matt Behnke
'  Created: 9/19/01
'  Description: formats the sheet to fit on one page wide (legal size paper)
'               adds header and footer to each sheet and sets orientation to landscape
'  inputs: none
'  Outputs: none
'--------------------------------------------------------------------------
Sub formatSheetForPrint()
'column heading                                          (R11.3)
    With ActiveSheet.PageSetup
'       .PrintTitleRows = "$1:$1"
 '      .PrintTitleColumns = ""
    End With
'    ActiveSheet.PageSetup.PrintArea = "$A$1:$Y$203"
    With ActiveSheet.PageSetup
 '      .LeftHeader = ""
  '     .CenterHeader = "&A in &F"                          '(R11.4)
   '     .RightHeader = ""
    '    .LeftFooter = "&D"                                 '(R11.5)
     '    .CenterFooter = "Page &P of &N"
      '    .RightFooter = ""
       '    .LeftMargin = Application.InchesToPoints(0.75)
       '    .RightMargin = Application.InchesToPoints(0.75)
        '    .TopMargin = Application.InchesToPoints(1)
'       .BottomMargin = Application.InchesToPoints(1)
 '      .HeaderMargin = Application.InchesToPoints(0.5)
  '     .FooterMargin = Application.InchesToPoints(0.5)
   '     .PrintHeadings = False
    '    .PrintGridlines = True
     '    .PrintComments = xlPrintNoComments
      '    .CenterHorizontally = False
       '    .CenterVertically = False
        '    .Orientation = xlLandscape                     '(R11.6)
   '     .Draft = False
    '    .PaperSize = xlPaperLetter                         '(R11.1)
     '    .FirstPageNumber = xlAutomatic
      '    .Order = xlDownThenOver
       '    .BlackAndWhite = False
      '    .Zoom = False
       '    .FitToPagesWide = 1                            '(R11.2)
        '    .FitToPagesTall = 99
    End With
End Sub 'format sheet for print

Sub rSquaredtest()

Call rSquaredSheet("World")
Call rSquaredSheet("A_Band")
Call rSquaredSheet("B_Band")
Call rSquaredSheet("C_Band")
Call rSquaredSheet("D_Band")
End Sub

'--------------------------------------------------------------------------------
' rSquaredSheet
' author: Matt Behnke
```

```
' created 1/3/02
'   creates a new sheet that stores the R*R values of the graphs:
'       number of publications over time and cumulative entropy over time
'   Uses the information stored in the affiliation Summary sheets..
'-------------------------------------------------------------------------------

Sub rSquaredSheet(ByVal band As String)

    Sheets.Add After:=Worksheets(Worksheets.Count)
    'Sheets(Worksheets.Count).Select

    currSheet = ActiveSheet.Name
    Sheets(currSheet).Select

    'set the source of the data
    If band = "World" Then
        source = "Affiliation_Summary"
    Else
        source = "Affiliation_Summary_" & band
    End If

    numRows = CountRows(source, 1)

    'fill in the header information of the rsquared sheet
    Call rSquaredSheetHeader(numRows - 3, currSheet, band)

    'determine startrow
    startRow = 4
    While Not (i > 0#)   'if the first time step's value is zero find the step that isn't 0
        i = Sheets(source).Cells(startRow, 3).Value
        If Not (i > 0) Then
            startRow = startRow + 1
        End If
    Wend

    counter = 6
    For i = startRow To numRows
        'get the month
        j = 1
        While found = False
            testchar = Sheets(source).Cells(i, 2).Characters(j, 1).Text
            If testchar = "/" Then
                found = True
            Else
            j = j + 1
            End If
        Wend
        'month ends at j

        'get the year
        currentMonth = Sheets(source).Cells(i, 2).Characters(1, j - 1).Text
        If currentMonth < 10 Then
            theYear = Sheets(source).Cells(i, 2).Characters(5, 2).Text
        Else
            theYear = Sheets(source).Cells(i, 2).Characters(6, 2).Text
        End If
```

```
        'test the year to see if different from before
        If Not theYear = previousYear Then
            startGraphRange = i
            startStep = Sheets(source).Cells(i, 1)
            Sheets(currSheet).Cells(counter, 2) = startStep
            If theYear > 50 Then
                Sheets(currSheet).Cells(counter, 1) = "19" & theYear
            Else
                Sheets(currSheet).Cells(counter, 1) = "20" & theYear
            End If

            Call rSquaredGraph(currSheet, startGraphRange, counter, source)
            counter = counter + 1
        End If

        'update previous year value
        previousYear = theYear
        found = False
    Next i

    Sheets(currSheet).Name = band & "_rSquared_Power"
End Sub 'rSquaredSheet
```

```
'-------------------------------------------------------------------
' rSquaredHeader
' Author: Matt Behnke
' Created 1 / 3 / 2002
' creates the header columns and formatting for the rsquared sheet
'
'-------------------------------------------------------------------
Sub rSquaredSheetHeader(ByVal numSteps As Integer, ByVal rSquaredSheetName As String, ByVal band
As String)

    Sheets(rSquaredSheetName).Select

    Range("A1").Select
    ActiveCell.FormulaR1C1 = "R-Squared Values for Ada, " & band
    Range("A3").Select
    ActiveCell.FormulaR1C1 = numSteps & " total steps, " & numSteps / 12 & " years."
    Range("A4").Select
    ActiveCell.FormulaR1C1 = "Number of Publications"

    Range("A5").Select
    ActiveCell.FormulaR1C1 = "Year"
    Range("B5").Select
    ActiveCell.FormulaR1C1 = "Starting Step"
    Range("C5").Select
    ActiveCell.FormulaR1C1 = "R_squared"
    Range("D5").Select
    ActiveCell.FormulaR1C1 = "Equation"

    Columns("A:J").Select
    Selection.ColumnWidth = 13.29

    Columns("D:D").Select
    Selection.ColumnWidth = 18

    Columns("F:F").Select
    Selection.ColumnWidth = 18

    Range("C5:D5").Select
    Selection.Copy

    Range("E5").Select
    ActiveSheet.Paste

    Range("E4").Select
    Application.CutCopyMode = False
    ActiveCell.FormulaR1C1 = "Entropy"

    Range("A5:I5").Select
    Selection.Borders(xlDiagonalDown).LineStyle = xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    Selection.Borders(xlEdgeLeft).LineStyle = xlNone
    Selection.Borders(xlEdgeTop).LineStyle = xlNone
    With Selection.Borders(xlEdgeBottom)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = xlAutomatic
```

```
    End With
    Selection.Borders(xlEdgeRight).LineStyle = xlNone
    Selection.Borders(xlInsideVertical).LineStyle = xlNone

    Columns("C:C").Select
    With Selection.Borders(xlEdgeLeft)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = xlAutomatic
    End With

    Columns("E:E").Select
    With Selection.Borders(xlEdgeLeft)
        .LineStyle = xlContinuous
        .Weight = xlThin
        .ColorIndex = xlAutomatic
    End With

    Range("A1").Select
    Selection.Font.Bold = True
End Sub 'rsquaredHeader


'-----------------------------------
' Sub: rSquaredGraph
' author: Matt Behnke
' Date: 1/3/2002
' uses a graph to determine the rsqurared value and equation ..
'  uses affiliation summary sheets
'-----------------------------------------------
Sub rSquaredGraph(ByVal rSquaredSheetName As String, ByVal startGraphRange As Integer, ByVal
counter As Integer, ByVal source As String)

'trendType = xlLinear
trendType = xlPower

'number of publications
    Charts.Add
    chartName = ActiveChart.Name
    ActiveChart.ChartType = xlLineMarkers
    ActiveChart.SetSourceData source:=Sheets(source).Range( _
        "C" & startGraphRange & ":C255"), PlotBy:=xlColumns

    With ActiveChart
        .HasTitle = False
        .Axes(xlCategory, xlPrimary).HasTitle = False
        .Axes(xlValue, xlPrimary).HasTitle = False
    End With


    ActiveChart.SeriesCollection(1).Select
    ActiveChart.SeriesCollection(1).Trendlines.Add(Type:=trendType, Forward:=0, _
        Backward:=0, DisplayEquation:=True, DisplayRSquared:=True).Select

'get trendline rsq and equation for num publications
    ActiveChart.SeriesCollection(1).Select
```

```
    With ActiveChart.SeriesCollection(1).Trendlines(1)
      trendEq = .DataLabel.Text
    End With

    Sheets(source).Select
    firstPartEq = firstPartTrendEq(trendEq)
    secondPartEq = secondPartTrendEq(trendEq)
    rSquared = rSquaredTrendEq(trendEq)

    If trendType = xlPower Then
       Sheets(rSquaredSheetName).Cells(counter, 4) = "y=" & firstPartEq & "x^" & secondPartEq
    Else
       Sheets(rSquaredSheetName).Cells(counter, 4) = "y=" & firstPartEq & "x + " & secondPartEq
    End If
    Sheets(rSquaredSheetName).Cells(counter, 3) = rSquared

'entropy
    Sheets(chartName).Select
    If Not Sheets(source).Cells(startGraphRange, 6) > 0 And trendType = xlPower Then

          Sheets(rSquaredSheetName).Cells(counter, 5) = "N/A due to zero entropy"

    Else
       ActiveChart.SetSourceData source:=Sheets(source).Range( _
          "F" & startGraphRange & ":F255"), PlotBy:=xlColumns

    'get trendline rsq and equation for entropy
       ActiveChart.SeriesCollection(1).Select
       With ActiveChart.SeriesCollection(1).Trendlines(1)
          trendEq = .DataLabel.Text
       End With

       Sheets(source).Select
       firstPartEq = firstPartTrendEq(trendEq)
       secondPartEq = secondPartTrendEq(trendEq)
       rSquared = rSquaredTrendEq(trendEq)

       If trendType = xlPower Then
          Sheets(rSquaredSheetName).Cells(counter, 6) = "y=" & firstPartEq & "x^" & secondPartEq
       Else
          Sheets(rSquaredSheetName).Cells(counter, 6) = "y=" & firstPartEq & "x + " & secondPartEq
       End If
       Sheets(rSquaredSheetName).Cells(counter, 5) = rSquared

    End If


'delete chart
    Sheets(rSquaredSheetName).Select
    Application.DisplayAlerts = False
    Sheets(chartName).Delete
    Application.DisplayAlerts = True


End Sub 'rSquaredGraph
'--------------------------------------------------------------------------
```

```
'  Function: rSquaredTrendEq
'  Author: Matt Behnke
'  Created: 1/3/02
'  Description: extracts the rSquared value of a trendline equation
'  inputs:   trendline equation
'  Outputs: firstpart of trendline equation
'--------------------------------------------------------------------------
Function rSquaredTrendEq(ByVal trendlineEq As String) As Double

    tempStorage = Cells(1, 1)
    Cells(1, 1) = trendlineEq

    i = 1
    While found = False
       testchar = Cells(1, 1).Characters(i, 1).Text
       If testchar = "R" Then
          found = True
       Else
       i = i + 1
       End If
     Wend

    'i = location of R
    'secondpart starts at character i plus 5..
    'num of characters = location(x) - 5
    'extract 5 characters..

    rSquaredTrendEq = Cells(1, 1).Characters(i + 5, 6).Text
    Cells(1, 1) = tempStorage

End Function ' rSquaredTrendEqu


'--------------------------------------------------------------------------
'  Function: CountRows
'  Author: ? Revised by: Matt Behnke
'  Created: ?
'  Revised: 9/10/01
'  Description: Counts the rows in the suppiled worksheet and column number
'  inputs:   sheetName - name of the sheet to count the rows in
'            colNum - number of the column to count rows in
'  Outputs: number of rows as a double
'--------------------------------------------------------------------------
Function CountRows(ByVal sheetName As String, ByVal colNum As Integer) As Double
On Error Resume Next
Dim currCell As Range, rowNum As Double

  Sheets("" & sheetName).Select

  If IsNumeric(colNum) Then
  Else
    colNum = 1
  End If

  rowNum = 1
  Set currCell = ActiveSheet.Cells(rowNum, colNum)
```

```
  Do While currCell.Value <> ""
    rowNum = rowNum + 1
    Set currCell = ActiveSheet.Cells(rowNum, colNum)
  Loop
  CountRows = rowNum - 1
End Function 'CountRows
'--------------------------------------------------------------------------
'  Function: CountCols
'  Author: ? Revised by: Matt Behnke
'  Created: ?
'  Revised: 9/10/01
'  Description: Counts the rows in the suppiled worksheet and column number
'  inputs:   sheetName - name of the sheet to count the columns in
'            rowNum - number of the row to count columns in
'  Outputs: number of columns as a double
'--------------------------------------------------------------------------
Function CountCols(ByVal sheetName As String, ByVal rowNum As Integer) As Integer
On Error Resume Next
Dim currCell As Range, colNum As Integer

  Sheets("" & sheetName).Select

  If IsNumeric(rowNum) Then
  Else
    rowNum = 1
  End If
  colNum = 1
  Set currCell = ActiveSheet.Cells(rowNum, colNum)
  Do While currCell.Value <> ""
    colNum = colNum + 1
    Set currCell = ActiveSheet.Cells(rowNum, colNum)
  Loop
  CountCols = colNum - 1
End Function 'CountCols
'--------------------------------------------------------------------------
'  Function: firstPartTrendEq
'  Author: Matt Behnke
'  Created: 11/13/01
'  Description: extracts the first part of the given POWER trendline equation, works w/ linear
'  inputs:   trendline equation
'  Outputs: firstpart of trendline equation
'--------------------------------------------------------------------------
Function firstPartTrendEq(ByVal trendlineEq As String) As Double

    tempStorage = Sheets(dataSheet).Cells(1, 1)
    Sheets(dataSheet).Cells(1, 1) = trendlineEq

    i = 1
    attempt = 1
    While found = False
       testchar = Sheets(dataSheet).Cells(1, 1).Characters(i, 1).Text
       If testchar = "x" Then
          found = True
       Else
       i = i + 1
          If i > 100 Then
```

```
            i = 1
            attempt = attempt + 1
          End If
          If attempt = 5 Then
            MsgBox ("Trend line read error! string: " & trendlineEq)
          End If
        End If
      Wend

    'i = location of x
    'firstpart = starts at character 5
    'num of characters = location(x) - 5

    firstPartTrendEq = Sheets(dataSheet).Cells(1, 1).Characters(5, i - 5).Text
    Sheets(dataSheet).Cells(1, 1) = tempStorage

End Function ' first part trendline
'--------------------------------------------------------------------------
'  Function: secondPartTrendEq
'  Author: Matt Behnke
'  Created: 11/13/01
'  Description: extracts the second part of the given POWER trendline equation
'  inputs:   trendline equation
'  Outputs: firstpart of trendline equation
'--------------------------------------------------------------------------
Function secondPartTrendEq(ByVal trendlineEq As String) As Double

    tempStorage = Sheets(dataSheet).Cells(1, 1)
    Sheets(dataSheet).Cells(1, 1) = trendlineEq

    i = 1
    While found = False
      testchar = Sheets(dataSheet).Cells(1, 1).Characters(i, 1).Text
      If testchar = "x" Then
        found = True
      Else
      i = i + 1
      End If
    Wend

    'i = location of x
    'secondpart starts at character i plus 1..
    'num of characters = location(x) - 5
    'extract 5 characters..

    secondPartTrendEq = Sheets(dataSheet).Cells(1, 1).Characters(i + 1, 5).Text
    Sheets(dataSheet).Cells(1, 1) = tempStorage

End Function ' secondPart eq
'--------------------------------------------------------------------------
'  Function: secondPartLinearTrendEq
'  Author: Matt Behnke
'  Created: 2/5/02
'  Description: extracts the second part of the given linear trendline equation
'  inputs:   trendline equation
'  Outputs: secondPart of trendline equation
```

```
'-------------------------------------------------------------------------
Function secondPartLinearTrendEq(ByVal trendlineEq As String) As Double

   tempStorage = Sheets(dataSheet).Cells(1, 1)
   Sheets(dataSheet).Cells(1, 1) = trendlineEq

   i = 1
   While found = False
      testchar = Sheets(dataSheet).Cells(1, 1).Characters(i, 1).Text
      If testchar = "x" Then
         found = True
      Else
      i = i + 1
      End If
    Wend

   'i = location of x
   'secondpart starts at character i plus 1..
   'num of characters = location(x) +6
   ' 4.143x + 2.4441
   '       ^^^^^^^^^
   'extract 9 characters..

   secondPartLinearTrendEq = Sheets(dataSheet).Cells(1, 1).Characters(i + 1, 9).Text
   Sheets(dataSheet).Cells(1, 1) = tempStorage

End Function ' secondPart eq

'-------------------------------------------------------------------------
'  Function: firstPartPolyTrendEq
'  Author: Matt Behnke
'  Created: 2/1/02
'  Description: extracts the first part of the given trendline equation
'               form ax2 + bx + c
'  inputs:   trendline equation
'  Outputs: firstpart of trendline equation
'-------------------------------------------------------------------------
Function firstPartPolyTrendEq(ByVal trendlineEq As String) As Double

   tempStorage = Sheets(dataSheet).Cells(1, 1)
   Sheets(dataSheet).Cells(1, 1) = trendlineEq

   i = 1
   While found = False
      testchar = Sheets(dataSheet).Cells(1, 1).Characters(i, 1).Text
      If testchar = "x" Then
         found = True
      Else
      i = i + 1
      End If
    Wend

   'i = location of x
   'firstpart = starts at character 5
   'num of characters = location(x) - 5
```

```
    firstPartPolyTrendEq = Sheets(dataSheet).Cells(1, 1).Characters(5, i - 5).Text
    Sheets(dataSheet).Cells(1, 1) = tempStorage

End Function ' first part poly order - 2 trendline
'--------------------------------------------------------------------------
'  Function: secondPartPolyTrendEq
'  Author: Matt Behnke
'  Created: 2/1/02
'  Description: extracts the second part of a second order polygonal trendline the given trendline equation
'  inputs:   trendline equation
'  Outputs: firstpart of trendline equation
'--------------------------------------------------------------------------
Function secondPartPolyTrendEq(ByVal trendlineEq As String) As Double

    tempStorage = Sheets(dataSheet).Cells(1, 1)
    Sheets(dataSheet).Cells(1, 1) = trendlineEq

    i = 1
    While found = False
       testchar = Sheets(dataSheet).Cells(1, 1).Characters(i, 1).Text
       If testchar = "x" Then
          found = True
       Else
       i = i + 1
       End If
     Wend

    j = i + 1
    While found2 = False
       testchar = Sheets(dataSheet).Cells(1, 1).Characters(j, 1).Text
       If testchar = "x" Then
          found2 = True
       Else
       j = j + 1
       End If
     Wend

    'i = location of first x
    'j = location of second x
    'secondpart starts at character i plus 5..
    '     i12345    j1234
    '1.0000x2 + 2.001x + 8.878
    'num of characters = j - i + 5


    secondPartPolyTrendEq = Sheets(dataSheet).Cells(1, 1).Characters(i + 5, j - (i + 5)).Text
    Sheets(dataSheet).Cells(1, 1) = tempStorage

End Function ' secondPart poly eq

'--------------------------------------------------------------------------
'  Function: thirdPartPolyTrendEq
'  Author: Matt Behnke
'  Created: 2/1/02
'  Description: extracts the third part of a second order polygonal trendline the given trendline equation
'  inputs:   trendline equation
```

```
'  Outputs: firstpart of trendline equation
'--------------------------------------------------------------------------
Function thirdPartPolyTrendEq(ByVal trendlineEq As String) As Double

    tempStorage = Sheets(dataSheet).Cells(1, 1)
    Sheets(dataSheet).Cells(1, 1) = trendlineEq

    i = 1
    While found = False
       testchar = Sheets(dataSheet).Cells(1, 1).Characters(i, 1).Text
       If testchar = "x" Then
          found = True
       Else
       i = i + 1
       End If
     Wend

    j = i + 1
    While found2 = False
       testchar = Sheets(dataSheet).Cells(1, 1).Characters(j, 1).Text
       If testchar = "x" Then
          found2 = True
       Else
       j = j + 1
       End If
     Wend

    'i = location of first x
    'j = location of second x
    'secondpart starts at character i plus 5..
    '     i12345    j1234
    '1.0000x2 + 2.001x + 8.878

    'third part starts at character j plus 4..
    'extract 5 characters..

    thirdPartPolyTrendEq = Sheets(dataSheet).Cells(1, 1).Characters(j + 4, 5).Text
    Sheets(dataSheet).Cells(1, 1) = tempStorage

End Function ' thirdPart poly eq

'--------------------------------------------------------------------------
'  Function: cols
'  Author: Matt Behnke
'  Created: 9/11/01
'  Description: changes column number into a letter.
'  inputs:   columnNumber
'  Outputs: column letter
'--------------------------------------------------------------------------
Function col(ByVal columnNumber As Integer) As String

  Select Case columnNumber
    Case 1
       col = "A"
    Case 2
       col = "B"
```

```
 Case 3
    col = "C"
 Case 4
    col = "D"
 Case 5
    col = "E"
 Case 6
    col = "F"
          .
          .
          .
          .
Case 236
    col = "IB"
 Case 237
    col = "IC"
 Case 238
    col = "ID"
 Case 239
    col = "IE"
 Case 240
    col = "IF"
 Case 241
    col = "IG"
 Case 242
    col = "IH"
 Case 243
    col = "II"
 Case 244
    col = "IJ"
 Case 245
    col = "IK"
 Case 246
    col = "IL"
 Case 247
    col = "IM"
 Case 248
    col = "IN"
 Case 249
    col = "IO"
 Case 250
    col = "IP"
 Case 251
    col = "IQ"
 Case 252
    col = "IR"
 Case 253
    col = "IS"
 Case 254
    col = "IT"
 Case 255
    col = "IU"

 Case others
    col = "Z"
End Select
```

```
End Function 'col


Public Function Update_Mathcad_Band_Stats(ByVal mathcad_sheet_name As String, _
   ByVal data_sheet_name As String, ByVal start_cell_x As Variant, _
   ByVal start_cell_y As Variant, ByVal num_rows As Integer, ByVal fit_type As Integer, _
   ByVal tolerance As Double) As Variant


'-------------------------------------------------------------------------
'  Function: Update_Mathcad_Band_Stats
'  Author: Aaron Micyus
'  Last Modified: 12/05/2001
'  Description: Given location information for input data this subroutine
'   passes data to embedded mathcad object for processing and returns
'   obtained values
'  inputs:
'      mathcad_sheet_name :    This is the name of the sheet the embedded
'                              object is in
'      data_sheet_name :   This is the name of the sheet we will obtain data
'                          from
'      start_cell_x    :   This is the cell location we start getting x data from
'      start_cell_y    :   This is the cell location we start getting y data from
'      num_rows        :   This is the number of data rows we have
'      fit_type        :   integer value corresponding to fit type to return
'                          0 - Hyperbolic 3 parameter (k,p,r)
'                          1 - Exponential 3 parameter (k,p,r)
'                          2 - Power 2 parameter (b,m)
'      tolerance       :   tolerance value for fit
'
'  Outputs:
'      array           :   returned array will hold calculated k,p,r values
'                          [1] element one : k
'                          [2] element two : r
'                          [3] element three: p
'                          [4] element four : Rsquared value
'-------------------------------------------------------------------------


    'VARS
    Dim Mathcad As Object                       'our interface to the Mathcad
                                                'embedded object
    Dim data_x_real, data_x_imag As Variant     'vars for real and imaginary
                                                'components of x data
    Dim data_y_real, data_y_imag As Variant     'vars for real and imaginary
                                                'components of y data
    Dim tolerance_real, tolerance_imag As Variant   'vars for real and imaginary
                                                'components of tolerance

    Dim k_real, k_imag As Variant               'vars for real and imag
                                                'k values from mathcad
    Dim r_real, r_imag As Variant               'vars for real and imag
                                                'r values from mathcad
    Dim p_real, p_imag As Variant               'vars for real and imag
                                                'p values from mathcad
```

```
    Dim b_real, b_imag As Variant                   'vars for real and imag
                                        'b values from mathcad
    Dim m_real, m_imag As Variant                   'vars for real and imag
                                        'm values from mathcad
    Dim rsquared_real, rsquared_imag As Variant     'vars for real and imag
                                        'r squared values from mathcad

    Dim current_char_position As Variant            'temp var to hold position in string
    Dim range_x, range_y As Variant                 'vars for calculated ranges

    Dim fit_results(4) As Variant                   'array to hold returned fit data from mathcad



    'initialize embedded mathcad
    Call Register_Mathcad_OLE(mathcad_sheet_name)

    'activate the sheet with the embedded mathcad object
    Worksheets(mathcad_sheet_name).Activate

    'get object reference
    Set Mathcad = ActiveSheet.OLEObjects(1).Object

    'activate the sheet with the data
    Worksheets(data_sheet_name).Activate

    """""construct the x value range
    'this temp variable holds current position in string we are parsing through
    current_char_position = 1

    'traverse the string until we find a numeric character
    While Not IsNumeric(Mid(start_cell_x, current_char_position, 1))

        current_char_position = current_char_position + 1

    Wend

    'calculate range string for x
    range_x = start_cell_x & ":" & Left(start_cell_x, 1)
    range_x = range_x & (Right(start_cell_x, (Len(start_cell_x) - current_char_position + 1)) + num_rows -
1)

    """""""now construct y value range
    'this temp variable holds current position in string we are parsing through
    current_char_position = 1

    'traverse the string until we find a numeric character
    While Not IsNumeric(Mid(start_cell_y, current_char_position, 1))

        current_char_position = current_char_position + 1

    Wend

    'calculate range string for y
    range_y = start_cell_y & ":" & Left(start_cell_y, 1)
```

```
    range_y = range_y & (Right(start_cell_y, (Len(start_cell_y) - current_char_position + 1)) + num_rows -
1)

    'set ranges for input data for mathcad
    data_x_real = ActiveSheet.Range(range_x).Value
    data_x_imag = Empty
    data_y_real = ActiveSheet.Range(range_y).Value
    data_y_imag = Empty


    tolerance_real = tolerance              'obtained from parameter
    tolerance_imag = Empty

    'import values into mathcad
    Call Mathcad.SetComplex("X_in", data_x_real, data_x_imag)
    Call Mathcad.SetComplex("Y_in", data_y_real, data_y_imag)
    Call Mathcad.SetComplex("eTOL", tolerance_real, tolerance_imag)

    'have mathcad recalculate sheet
    Call Mathcad.Recalculate

    If fit_type = HYP3_FIT Then

        'get values from mathcad for excel
        Call Mathcad.GetComplex("out0", k_real, k_imag)
        Call Mathcad.GetComplex("out1", r_real, r_imag)
        Call Mathcad.GetComplex("out2", p_real, p_imag)
        Call Mathcad.GetComplex("out3", rsquared_real, rsquared_imag)

        'fill array with results
        fit_results(1) = k_real
        fit_results(2) = r_real
        fit_results(3) = p_real
        fit_results(4) = rsquared_real

    ElseIf fit_type = EXP3_FIT Then

        'get values from mathcad for excel
        Call Mathcad.GetComplex("out4", k_real, k_imag)
        Call Mathcad.GetComplex("out5", r_real, r_imag)
        Call Mathcad.GetComplex("out6", p_real, p_imag)
        Call Mathcad.GetComplex("out7", rsquared_real, rsquared_imag)

        'fill array with results
        fit_results(1) = k_real
        fit_results(2) = r_real
        fit_results(3) = p_real
        fit_results(4) = rsquared_real

    ElseIf fit_type = POW2_FIT Then

        'get values from mathcad for excel
        Call Mathcad.GetComplex("out8", b_real, b_imag)
        Call Mathcad.GetComplex("out9", m_real, m_imag)
        Call Mathcad.GetComplex("out10", rsquared_real, rsquared_imag)
```

```
        'fill array with results
        fit_results(1) = b_real
        fit_results(2) = m_real
        fit_results(3) = Empty
        fit_results(4) = rsquared_real

    End If

    Update_Mathcad_Band_Stats = fit_results

'end of Update_Mathcad_Band_Stats
End Function

Public Function Register_Mathcad_OLE(ByVal mathcad_sheet_name As String)

'
' register_mathcad_ole Macro
' opens embedded mathcad document in order for system to recognize it for future macro
'

'
    Sheets(mathcad_sheet_name).Select
    Range("A1").Activate
    ActiveSheet.Shapes("Object 1").Select
    Selection.Verb Verb:=xlPrimary
    Range("A1").Activate
    'Sheets("A_Band_Stats").Select
End Function

'---------------------------------------------------------------------------
'  sub: fillMonthsCol
'  Author: Matt Behnke
'  Created: 12/11/01
'  Description: fills in the months if they are missing.. Inserts a column used for a matrix sheet
'               this doesnt work because there are not enough columns
'  inputs:  sheetName
'  Outputs:
'---------------------------------------------------------------------------
Sub fillMonthsCol() ' ByVal sheetName As String)

    sheetName = ActiveSheet.Name
    numColumns = CountCols(sheetName, 1)
    Dim theMonth As Integer

    counter = 1
    monthCounter = "" & counter
    For i = 4 To numColumns

        j = 1
        While found = False
            testchar = Cells(1, i).Characters(j, 1).Text
            If testchar = "/" Then
                found = True
            Else
            j = j + 1
            End If
```

```
        Wend
        'month ends at j

        currentMonth = Cells(1, i).Characters(1, j - 1).Text
        If currentMonth < 10 Then
           restofDate = Cells(1, i).Characters(2, 5).Text
        Else
           restofDate = Cells(1, i).Characters(3, 5).Text
        End If
        theMonth = currentMonth
        If theMonth > monthCounter Then
           While theMonth > monthCounter
              Range(Cells(1, i), Cells(1, i)).Select
              Selection.EntireColumn.Insert

              'copy previous column
              Columns(col(i - 1) & ":" & col(i - 1)).Select
              Selection.Copy
              Columns(col(i) & ":" & col(i)).Select
              ActiveSheet.Paste
              Cells(1, i) = "'" & monthCounter & restofDate

              counter = counter + 1
              If counter = 13 Then
                 counter = 1
              End If
              monthCounter = "" & counter
              i = i + 1
              numColumns = numColumns + 1
           Wend
        End If


        counter = counter + 1
        If counter = 13 Then
           counter = 1
        End If
        monthCounter = "" & counter
        found = False
     Next i
     Columns("D:D").ColumnWidth = 6.29
End Sub ' fill months col
```

```
'--------------------------------------------------------------------------
'  sub: fillMonthsRow
'  Author: Matt Behnke
'  Created: 12/11/01
'  Description: fills in the months if they are missing.. Inserts a row
'               works on lists.... must run 2 - 3 times to ensure all filled..
'  inputs:  sheetName
'  Outputs:
'--------------------------------------------------------------------------
Sub fillMonthsRow(ByVal sheetName As String, ByVal startRow As Integer) ' ByVal sheetName As
String)

   Dim numRows As Integer
   Dim theYear As Integer
   Dim theMonth As Integer
   Dim previousYear As Integer
   Dim previousMonth As Integer
   Dim monthCounter As String
   Dim i As Integer

   Sheets(sheetName).Select
   numColumns = CountCols(sheetName, 1)
   numRows = CountRows(sheetName, 1)



   i = startRow

   counter = 1
   monthCounter = "" & counter
   While Not i = numRows + 1

      j = 1
      While found = False
         testchar = Cells(i, 2).Characters(j, 1).Text
         If testchar = "/" Then
            found = True
         Else
         j = j + 1
           If j > 100 Then
               j = 1
               test = test + 1
               If test > 5 Then
                  shit = 1
                  Exit Sub
               End If
           End If
         End If
      Wend
      'month ends at j

      currentMonth = Cells(i, 2).Characters(1, j - 1).Text
      If currentMonth < 10 Then
         theYear = Cells(i, 2).Characters(5, 4).Text
      Else
         theYear = Cells(i, 2).Characters(6, 4).Text
```

```
End If

theMonth = currentMonth

If i = startRow Then
  monthCounter = theMonth
  counter = theMonth
End If

If theYear = previousYear + 1 And theMonth = 12 And previousMonth = 11 Then
   'case ex: 11/1/1988 -- > 12/1/1989
      'need to fill in 12/1/1987 --> 1/1/1989

      Range(Cells(i, 1), Cells(i, 1)).Select
         Selection.EntireRow.Insert
      Rows(i - 1 & ":" & i - 1).Select
      Selection.Copy
      Rows(i & ":" & i).Select
      ActiveSheet.Paste
      Cells(i, 2) = "'12/1/" & previousYear
      Cells(i, 1) = i - startRow + 1
      i = i + 1
      i = i - 1
      numRows = numRows + 1
End If


   If theMonth < monthCounter Then ' And monthCounter < 12 Then ' a new year has begun before a
previous year finished..
      Do While Not theMonth = monthCounter
         If i = startRow Then

         Else
            If monthCounter < 11 Then
               restofDate = Cells(i - 1, 2).Characters(2, 7).Text
              '  theYear = Cells(1, i).Characters(5, 4).Text
            Else
               restofDate = Cells(i - 1, 2).Characters(3, 7).Text
              ' theYear = Cells(1, i).Characters(6, 4).Text
            End If



            Range(Cells(i, 1), Cells(i, 1)).Select
            Selection.EntireRow.Insert

         'copy previous row
            Rows(i - 1 & ":" & i - 1).Select
            Selection.Copy
            Rows(i & ":" & i).Select
            ActiveSheet.Paste
            Cells(i, 2) = "'" & monthCounter & restofDate
         End If 'i=startrow
         Cells(i, 1) = i - startRow + 1
         counter = counter + 1
         If counter = 13 Then
```

```
        counter = 1
      End If
      monthCounter = "" & counter
      i = i + 1
      numRows = numRows + 1

      previousYear = theYear
      previousMonth = theMonth

      If theMonth > monthCounter Then
        Exit Do
      End If
    Loop
End If


If theMonth > monthCounter Then
    While theMonth > monthCounter
      If i = startRow Then
        'do nothing
      Else
        If currentMonth < 10 Then
          restofDate = Cells(i, 2).Characters(2, 7).Text
        Else
          restofDate = Cells(i, 2).Characters(3, 7).Text
        End If
        Range(Cells(i, 1), Cells(i, 1)).Select
        Selection.EntireRow.Insert


      'copy previous row
        Rows(i - 1 & ":" & i - 1).Select
        Selection.Copy
        Rows(i & ":" & i).Select
        ActiveSheet.Paste
        Cells(i, 2) = "'" & monthCounter & restofDate
      End If 'i=startrow

      Cells(i, 1) = i - startRow + 1
      counter = counter + 1
      If counter = 13 Then
        counter = 1
      End If
      monthCounter = "" & counter
      i = i + 1
      numRows = numRows + 1
      If i > 50 Then
      a = 1
      End If

    Wend
End If


Cells(i, 1) = i - startRow + 1
```

```
        counter = counter + 1
        If counter = 13 Then
           counter = 1
        End If
        monthCounter = "" & counter
        found = False
        i = i + 1

        previousYear = theYear
        previousMonth = theMonth

    Wend

End Sub ' fill months rows

Sub testFillMonthsRows()

            Call fillMonthsRow("World_Stats", 14)

End Sub

Sub fillMonthsRowTrigger()


If testBandA > 0 Then
    Call fillMonthsRow("A_Band_Stats", 14)

    Call fillMonthsRow("Affiliation_Summary_A_Band", 4)

End If

If testBandB > 0 Then

    Call fillMonthsRow("B_Band_Stats", 14)
    Call fillMonthsRow("Affiliation_Summary_B_Band", 4)

End If

If testBandC > 0 Then
    Call fillMonthsRow("C_Band_Stats", 14)

    Call fillMonthsRow("Affiliation_Summary_C_Band", 4)

End If

If testBandD > 0 Then
    Call fillMonthsRow("D_Band_Stats", 14)

    Call fillMonthsRow("Affiliation_Summary_D_Band", 4)

End If

Call fillMonthsRow("World_Stats", 14)


Call fillMonthsRow("Affiliation_Summary", 4)
```

```
Call fillMonthsRow("Entropy Summary", 4)

End Sub

'--------------------------------------------------------------------
' sub v_calc_v_psi_sheet()
' Author: Matt Behnke
' Created: 2/22/02
' Description: creates a sheet either by band or world that has the result of v and psi.
'              where v = (num records at an instance in step k)/(total num of records at step k)
'                        ------------------------------------------------------------------
'                       (num of authors at an instance in step k)/(total num of authors at step k)
'
'              psi (tasks per timestep on avg) = v / timestep
'
' inputs: band - the source..
'
'--------------------------------------------------------------------
Sub v_calc_v_psi_sheet(ByVal band As String)

    'Dim authorMatrixTotals As Variant
    'Dim affiliationMatrixTotals As Variant
    Dim sum_v_array As Variant

    'N_i_k = the number of records produced by affiliation i at timestep k
    'N_Total_k = the number of records produced by all affiliations at timestep k
    'P_i_k = the number of authors who published in affiliation i at timestep k
    'P_Total_k = the number of authors who published in all affiliations at timestep k

    authorMatrixTotals = Array()
    affiliationMatrixTotals = Array()

    Sheets.Add After:=Worksheets(Worksheets.Count)

    'Sheets(Worksheets.Count).Select

    ActiveSheet.Name = "v_calculation_" & band
    currentSheetName = ActiveSheet.Name

    'move the sheet so it is by related sheets
'    Sheets(currentSheetName).Move Before:=Sheets("" & band & "_Stats")

    If band = "World" Then
        affiliationMatrix = dataSheet
        authorMatrix = "Affiliation_authors"
    Else
        affiliationMatrix = "Affiliation_Cum_Dist_" & band
        authorMatrix = "Aff_Author_Cum_Dist_" & band
    End If

    numRowsInAffiliationMatrix = CountRows(affiliationMatrix, 1)
    numColsInAffiliationMatrix = CountCols(affiliationMatrix, 1)

    numRowsInAuthorMatrix = CountRows(authorMatrix, 1)
    numColsInAuthorMatrix = CountCols(authorMatrix, 1)
```

```
    'ReDim affiliationMatrixTotals(4 To numColsInAffiliationMatrix)
    'ReDim authorMatrixTotals(4 To numColsInAuthorMatrix)
    ReDim sum_v_array(4 To numColsInAffiliationMatrix)

    'headers
    Sheets(currentSheetName).Cells(1, 1) = " "
    Sheets(currentSheetName).Cells(1, 2) = " "
    Sheets(currentSheetName).Cells(3, 1) = "Time Step"
    Sheets(currentSheetName).Cells(3, 2) = "interval"
    Sheets(currentSheetName).Cells(3, 3) = "v"
    Sheets(currentSheetName).Cells(3, 4) = "psi"

    For i = 2 To numRowsInAuthorMatrix
       For j = 4 To numColsInAffiliationMatrix
          If i = 2 Then
             timeStep = j - 3
             interval = Sheets(authorMatrix).Cells(1, j)
             Sheets(currentSheetName).Cells(j, 1) = timeStep 'the timestep
             Sheets(currentSheetName).Cells(j, 2) = interval 'the timestep
          End If
          affiliationNameFromAuthorMatrix = Sheets(authorMatrix).Cells(i, 3)

          'find the row that contains the affiliation name from authors matrix in the affiliation matrix
          'sheet

          N_i_k_row = findStringInSheet(affiliationMatrix, affiliationNameFromAuthorMatrix, "C")

          'temp = Sheets(datasheet).Cells(1, 1)
          'Sheets(dataSheet).Cells(1, 1) = N_i_k_range
          'N_i_k_row = Sheets(dataSheet).Cells(1, 1).Characters(4, 5).Text
          'Sheets(dataSheet).Cells(1, 1) = temp

          If j > 4 Then 'find the values at that instance... not cumulative
             P_i_k = Sheets(authorMatrix).Cells(i, j) - Sheets(authorMatrix).Cells(i, j - 1)
             N_i_k = Sheets(affiliationMatrix).Cells(N_i_k_row, j) -
Sheets(affiliationMatrix).Cells(N_i_k_row, j - 1)
             N_total = Sheets(affiliationMatrix).Cells(numRowsInAffiliationMatrix + 4, j) _
                - Sheets(affiliationMatrix).Cells(numRowsInAffiliationMatrix + 4, j - 1)
             P_total = Sheets(authorMatrix).Cells(numRowsInAuthorMatrix + 4, j) _
                - Sheets(authorMatrix).Cells(numRowsInAuthorMatrix + 4, j - 1)
          Else
             P_i_k = Sheets(authorMatrix).Cells(i, j)
             N_i_k = Sheets(affiliationMatrix).Cells(N_i_k_row, j)
             N_total = Sheets(affiliationMatrix).Cells(numRowsInAffiliationMatrix + 4, j)
             P_total = Sheets(authorMatrix).Cells(numRowsInAuthorMatrix + 4, j)
          End If
          'calculate v
          '    where v = (num records at an instance in step k)/(total num of records at step k)
          '              ------------------------------------------------------------------
          '              (num of authors at an instance in step k)/(total num of authors at step k)
          If P_i_k And N_total > 0 Then
             v = (((N_i_k / N_total) / P_i_k) / P_total)
          Else
             v = 0
          End If
```

```
            sum_v_array(j) = sum_v_array(j) + v
            'debug*****8
               Sheets(currentSheetName).Cells(i + 2, j + 4) = sum_v_array(j)
               Sheets(currentSheetName).Cells(1, 2) = j
            'debug
         Next j
         Sheets(currentSheetName).Cells(1, 1) = i
      Next i

      For j = 4 To numColsInAffiliationMatrix
      'output array of sum v.. make cumulative
         timeStep = j - 3


         If j > 4 Then 'cumulative
            Sheets(currentSheetName).Cells(j, 3) = sum_v_array(j) + Sheets(currentSheetName).Cells(j - 1, 3)
         Else
            Sheets(currentSheetName).Cells(j, 3) = sum_v_array(j)
         End If
         psi = Sheets(currentSheetName).Cells(j, 3) / timeStep

         Sheets(currentSheetName).Cells(j, 4) = psi
      Next j

End Sub 'calc_v_psi_sheet

'---------------------------------------------------------------------
' sub clearArray()
' Author: Matt Behnke
' Created: 2/22/02
' Description: clears the values stored in an array.
'
' inputs: lowerBound - lowerbound of the array
'         upperBound - upperbound of the array
'         arrayName - the array
'
'---------------------------------------------------------------------
Sub clearArray(ByVal lowerBound As Integer, ByVal upperBound As Integer, ByVal arrayName As
Variant)

   For i = lowerBound To upperBound
      arrayName(i) = ""
   Next i

End Sub

'---------------------------------------------------------------------
' function LinearInterpolation()
' Author: Matt Behnke
' Created: 2/26/02
' Description: linear interpolation used to calculate missing data:
'               [ X_i - X_low]
'   Y_i = y_low + [------------] * (Y_hi - Y_low)
'               [X_hi - X_low]
'
```

```
' inputs: Y_low - the closest "real" value to the left of the missing value
'         Y_high - the closest "real" value to the right of the missing value
'         X_low - the closest time step that has data to the left of the missing value
'         X_high - the closest time step that has data to the right of the missing value
'         X_i - the time step that has the missing data..
' output: returns a value, Y_i, for the missing time step.
'---------------------------------------------------------------------
Function LinearInterpolation(ByVal Y_low As Double, ByVal Y_high As Double, ByVal X_low As
Integer, _
                ByVal X_high As Integer, ByVal X_i As Integer) As Double

   LinearInterpolation = Y_low + ((X_i - X_low) / (X_high - X_low)) * (Y_high - Y_low)

End Function




'---------------------------------------------------------------------
' sub: FillInMissingData()
' Author: Matt Behnke
' Created: 2/26/02
' Description: Stores a list of data in an array, traverses the array to find
'             points where the data doesn't change. In our case where nothing was added
'             due to lack of information (small holes in the dataset).
'             When an element that doesn't change is found a linearInterpolation is performed
'             to determine what the value should be.
'              the value is changed and marked in red. <-- not added?
'
' inputs: dataSheet - the source of the data.
'         columnNumber - the column that contains the data
'         startRow - the row number where the data starts
'         endRow - the row number where the data ends
'---------------------------------------------------------------------
Sub FillInMissingData(ByVal dataSheet As String, ByVal columnNumber As Integer, ByVal startRow As
Integer, _
                ByVal endRow As Integer)

   Dim dataArray As Variant
   dataArray = Array()

   numTimeSteps = endRow - startRow + 1

   ReDim dataArray(1 To numTimeSteps)

   ' numTimesteps (k) = 5
   ' arrayindex (a) = 0 to 5
   ' startrow = 3
   ' endrow = 7
   ' k a   row
   ' 1 1  3+0=3
   ' 2 2  3+1=4
   ' 3 3  3+2=5
   ' 4 4  3+3=6
   ' 5 5  3+4=7
```

```
   'populate the array with data
   For i = 1 To numTimeSteps
      dataArray(i) = Sheets(dataSheet).Cells(startRow + i - 1, columnNumber)
   Next i

   'analyze the array:
   For i = 1 To numTimeSteps
      If Not i = numTimeSteps Then
         currentValue = dataArray(i)
         nextValue = dataArray(i + 1)
         If currentValue = nextValue Then

            lowestDifferentValue = dataArray(i)              'Y_low
            lowestDifferentValuePosition = i                 'X_low
            For j = i + 2 To numTimeSteps 'scan the array to find the next higher value
               nextHigherValue = dataArray(j)                'Y_high
               If Not nextHigherValue = currentValue Then
                  nextHigherValuePosition = j                'X_high
                  Exit For 'j
               End If
            Next j

            For k = lowestDifferentValuePosition + 1 To nextHigherValuePosition - 1
            'now the next lower and higher values are known along with their positions,call
linearInterpolation
             'k = y_i
               dataArray(k) = LinearInterpolation(lowestDifferentValue, nextHigherValue, _
                              lowestDifferentValuePosition, nextHigherValuePosition, k)
            Next k
         End If 'currentvalue = nextValue
      End If 'not equal to numTimesteps
   Next i

   'output the array
   For i = 1 To numTimeSteps
      Sheets(dataSheet).Cells(startRow + i - 1, columnNumber) = dataArray(i)
   Next i

End Sub 'fill in missing data

'TEST FILL IN MISSING DATA
Sub testFillInMissing()

'tests the linear interpolation function..can also be used as an interface to the function...

dataSheet = InputBox("enter the name of the source sheet")
columnNumber = InputBox("enter column number")
rowStart = InputBox("enter the first row of data")
rowEnd = InputBox("enter the last row of data")

   Call FillInMissingData(dataSheet, columnNumber, rowStart, rowEnd)


End Sub
```

```
'--------------------------------------------------------------------------
'  CurveFit
'  Author: Erchuang (Al) Wang (original), Matt Behnke - converted to VB
'  Converted: 2/27/02
'  Description:
'             This program will fit a curve up to 10th degree polynomial
'             in the form of Y = a0 + a1*x + a2*x^2 + ... + a(n)*x^(n)
'             where n is the degree of the polynomial and 1>=n=<30
'             Reads in two lists of numbers, X & Y-values and performs the fit
'
'  inputs:  dataSheet - sheet with the source values
'           numValues - the number of values in the array
'           x - array of the x values
'           y - array of the y-values
'           degree  -  the degree of the polynomial
'  Outputs: CoefficientArray - the coefficients of the equation a0, a1, ... a(n)
'--------------------------------------------------------------------------
Function curveFit(ByVal numValues As Integer, _
              ByVal x As Variant, ByVal y As Variant, ByVal degree As Integer) As Variant

   'numValues = endRow - startRow + 1

   'variables
   Dim coefficientArray(64) As Variant                    'stores the results, max of 64 coeff..
   'Dim x As Variant                                     'a one dimension array for x values
   'Dim y As Variant                                     'a one dimension array for y values
   Dim cn(64) As Variant                                  '??????????????????????????

   Dim ar(64, 64) As Variant                              'a two dimension array
   Dim an(64, 64) As Variant                              'answer array

   Dim sum As Double
   Dim t As Double
   Dim d As Double
   Dim b As Double

   Dim j As Integer                                       'for loop counter
   Dim i As Integer                                       'for loop counter
   Dim m As Integer
   Dim n As Integer                                       '=numValues, number of data points
   Dim ii As Integer                                      'for loop counter
   Dim k As Integer                                       'for loop counter
   Dim nn As Integer
   Dim nd As Integer                                      '=degree, degree of poly

   n = numValues
   nd = degree
   m = nd + 1
   nn = m + 1

   'generate normal equation A and vector B of Ax=B
   For ii = 1 To n
      For j = 1 To m
         If j = 1 And x(ii) = 0# Then
            ar(ii, j) = 1#
         Else
```

```
            ar(ii, j) = x(ii) ^ (j - 1)
        End If
    Next j
Next ii

For k = 1 To m
    For ii = 1 To m
        sum = 0#
        For j = 1 To n
            sum = sum + ar(j, k) * ar(j, ii)
        Next j
        an(k, ii) = sum
    Next ii
Next k

For ii = 1 To m
    sum = 0#
    For j = 1 To n
        sum = sum + y(j) * ar(j, ii)
    Next j
    cn(ii) = sum
Next ii

'solve x vector of Ax=B where A=A'

For i = 1 To m
    an(i, nn) = cn(i)
Next i

For i = 1 To m
    k = i
    b = Abs(an(i, i))
    If b = 0# Then
        For j = i To m
            If b < Abs(an(j, ii)) Then
                b = Abs(an(j, i))
                k = j
            End If
        Next j

        For j = 1 To nn
            t = an(i, j)
            an(i, j) = an(k, j)
            an(k, j) = t
        Next j

    Else
        d = an(i, i)
    End If

    For j = 1 To nn
        an(i, j) = an(i, j) / d
    Next j

    For j = 1 To m
        b = an(j, i)
```

```
            For k = 1 To nn
               If Not j = i Then
                  an(j, k) = an(j, k) - an(i, k) * b
               End If
            Next k
         Next j
      Next i

      'put answers into the coefficient array

      For ii = 1 To m
         coefficientArray(ii) = an(ii, nn)
      Next ii

      curveFit = coefficientArray

End Function 'curvefit

Sub testFind()

'WORKS'...
"Phys. Dept., Kakatiya Univ., Warangal, India
With ActiveSheet.Range("A:A")
   Set C = .Find("3/1979", LookIn:=xlValues)
   If Not C Is Nothing Then
      firstAddress = C.Address
      MsgBox (firstAddress)
   End If
End With
'MsgBox (Search("Phys. Dept., Kakatiya Univ., Warangal, India"))

End Sub
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX O – ENTROPY LAMBDA MACRO CODE

```
'****************************************
' Macro: EntropyLambda
'   Description: Computes entropy based on lambda and lyapunov exponent
'               Shows "mindshare" by dividing beta by author
'
'****************************************

Dim currFilename As String
Dim betaPercentage As Double

Sub runEntropyLambda()

            Dim sourceSheet As String
            sourceSheet = "Affiliation_Summary"

            currFilename = Application.ActiveWorkbook.Name

            betaPercentage = 0.05


Call EntropyLambda(sourceSheet)
Call Correlate_S_Lambda("EntropyLambda")

End Sub

'-------------------------------------------------------------------------
'  Subroutine: entropyLambda
'  Author: Matt Behnke
'  Created: 9/19/01
'  Revised: 9/24 - added map k, k+1 stuff
'           2/14/02 - made it a separate macro that uses the "Affiliation_Summary" sheet as
'                     its input.
'  Description: 1) Creates a sheet called "entropy lambda" where lambda and the Lyapunov
'                  number is calculated based on formulas given in the requirements
'  inputs:   sourceSheet.. uses the affiliation summary sheet.

'  Outputs: none
'-------------------------------------------------------------------------
Sub EntropyLambda(ByVal sourceSheet)

  numRows = CountRows(sourceSheet, 1) 'get the num of rows in the source sheet
  startRange = 4

'create new sheet                                        '(R6.1)
  Sheets.Add
  currentName = ActiveSheet.Name
  Sheets("" & currentName).Name = ("EntropyLambda")
  currentName = ActiveSheet.Name

'set height of header row and standard column width
  Rows("1:1").RowHeight = 52                             '(R12.6)
  ActiveSheet.StandardWidth = 12                          '(R12.5)
```

```
'headers                                                '(R6.3)
  ActiveSheet.Cells(1, 1) = " "
  ActiveSheet.Cells(2, 1) = " "
  ActiveSheet.Cells(3, 1) = "TimeStep (k)"
  ActiveSheet.Cells(3, 2) = "month/year"
  ActiveSheet.Cells(3, 3) = "Records (cumulative sum)"
  ActiveSheet.Cells(3, 4) = "Entropy (S)"
  ActiveSheet.Cells(3, 5) = "S(k+1)"
  ActiveSheet.Cells(3, 6) = "du_(t-c)"
  ActiveSheet.Cells(3, 7) = "du_(t)"
  ActiveSheet.Cells(3, 8) = "B_y_10%"
  ActiveSheet.Cells(3, 9) = "Lambda_B10%_y"
  ActiveSheet.Cells(3, 10) = "beta"
  ActiveSheet.Cells(2, 10) = betaPercentage

'fill in column 5: Cum_K+1                               '(R6.10)
  For i = startRange To numRows
    Sheets(currentName).Cells(i, 1) = Sheets(sourceSheet).Cells(i, 1)
    Sheets(currentName).Cells(i, 2) = Sheets(sourceSheet).Cells(i, 2)
    Sheets(currentName).Cells(i, 3) = Sheets(sourceSheet).Cells(i, 3)
    Sheets(currentName).Cells(i, 4) = Sheets(sourceSheet).Cells(i, 8)
    Sheets(currentName).Cells(i, 5) = "=D" & i + 1
  Next i

'create the map of k, k+1 to get the trendline equation for          '(R6.11)
'calculating the lyanponuv exponent.
  Call CopyMessagesGraph(startRange, numRows, currentName)
  Call CopyMapEntropyKK_1(startRange, numRows, currentName)
  Call CopyEntropyLambdaChart(startRange, numRows, currentName, False)

' After the three graphs are called the following trendlines are placed onto the
' entropy lambda sheet:
' cells(1,6) contains the trendline equation of messages (N, Records)
' cells(1,7) contains the trendline equation of entropy
' cells(1,8) containt the tren..... of lambda

'get formula of trendline from entropy power trend graph
    trendEq_entropy = Sheets(currentName).Cells(1, 7)
    m_s = firstPartTrendEq(trendEq_entropy)
    b_s = secondPartTrendEq(trendEq_entropy)

'get formula of trendline from the messages graph
   trendEq_messages = Sheets(currentName).Cells(1, 6)
    firstPart_messages = firstPartTrendEq(trendEq_messages)
    secondPart_messages = secondPartTrendEq(trendEq_messages)

  For i = startRange + 1 To numRows
    stepK = Sheets(currentName).Cells(i, 1)
    Sheets(currentName).Cells(i, 6) = firstPart_messages 'the derivative of the msgs trendlineEq
    Sheets(currentName).Cells(i, 7) = (m_s * b_s) * (stepK ^ (b_s - 1))
    Sheets(currentName).Cells(i, 10) = "=J2"
    Sheets(currentName).Cells(i, 8) = "=D" & i & "-I" & i 'cells(i,4) - cells (i, 9)
    Sheets(currentName).Cells(i, 9) = "=((J" & i & "*F" & i & ")/((J" & i & "*F" & i & ")+G" & i &
"))^(1/3)"
                        '(cells(i,10) * cells(i, 6))/((cells(i, 10)*cells(i,6)) + cells(i,7)) ^ (1/3)
```

```
'     ((J*F)
'  ------------
'   ((J*F)+G))       ^(1/3)

  Next i

Call CopyEntropyLambdaChart(startRange, numRows, currentName, True)

'betaPercentage = findBestBeta(currentName, numRows - 3) not used but can to determine
'the optimal beta percentage-- not fully tested..

    Sheets(currentName).Cells(2, 10) = betaPercentage

With Charts("Entropy Lambda Chart").SeriesCollection(1).Trendlines(1)
          'put trendline equation onto source
          .DisplayEquation = True
          .DisplayRSquared = True
          Sheets(currentName).Cells(1, 8) = .DataLabel.Text
End With

End Sub 'entropyLambda


'--------------------------------------------------------------------------
'  Function: findBestBeta
'  Author: Matt Behnke
'  Created: 5/9/2002
'  Description: Finds the best beta value by trying values from .01 to .5
'               the results are stored in an array.
'  inputs:
'  Outputs: none
'--------------------------------------------------------------------------
Function findBestBeta(ByVal sourceSheet As String, ByVal biggestTimeStepValue As Integer)


    Dim bestR As Double
    Dim currentR As Double
    Dim currentLargestValue As Double
    Dim bestLargestValue As Double
    Dim theBestPercentage As Double


    '.1 % to 30%
    For i = 1 To 300
       'change the beta percentage
          Sheets(sourceSheet).Cells(2, 10) = i / 1000


       'get the new trenline equations
       With Charts("Entropy Lambda Chart").SeriesCollection(1).Trendlines(1)
          'put trendline equation onto source
          .DisplayEquation = True
          .DisplayRSquared = True
          Sheets(sourceSheet).Cells(1, 8) = .DataLabel.Text
       End With
```

```
        trendEq_lambda = Sheets(sourceSheet).Cells(1, 8)
        m_lambda = firstPartTrendEq(trendEq_lambda)
        b_lambda = secondPartTrendEq(trendEq_lambda)

        'get the current values
        currentR = rSquaredTrendEq(trendEq_lambda)
        currentLargestValue = m_lambda * biggestTimeStepValue ^ b_lambda

        'compare the values
        If currentLargestValue < 1# Then
           If currentR > bestR Then
              bestR = currentR
              theBestPercentage = i / 1000
           End If
        Else

        End If

    Next i

    findBestBeta = theBestPercentage

End Function

'---------------------------------------------------------------------------
'  Subroutine: Correlate_S_Lambda
'  Author: Matt Behnke
'  Created: 5/19/02
'  Description: 1) Creates a sheet called "entropy lambda" where lambda and the Lyapunov
'               number is calculated based on formulas given in
'  inputs:   sourceSheet.. uses the entropy lambda sheet created from EntropyLambda().

'  Outputs: none
'---------------------------------------------------------------------------
Sub Correlate_S_Lambda(ByVal sourceSheet As String)

    Dim sourceSheet2 As String                  'holds the name of the affiliation summary sheet
    sourceSheet2 = "Affiliation_Summary"


  numRows = CountRows(sourceSheet, 1) 'get the num of rows in the source sheet
  startRange = 4

'create new sheet                                        '(R6.1)
  Sheets.Add
  currentName = ActiveSheet.Name
  Sheets("" & currentName).Name = ("Correlate_S_Lambda")
  currentName = ActiveSheet.Name

'set height of header row and standard column width
  Rows("3:3").RowHeight = 52                                '(R12.6)
  ActiveSheet.StandardWidth = 12                            '(R12.5)

'headers                                                 '(R6.3)
  ActiveSheet.Cells(1, 1) = "m_s"
  ActiveSheet.Cells(2, 1) = "b_s"
```

```
  ActiveSheet.Cells(1, 3) = "m_lambda"
  ActiveSheet.Cells(2, 3) = "b_lambda"
  ActiveSheet.Cells(3, 1) = "TimeStep (k)"
  ActiveSheet.Cells(3, 2) = "month/year"
  ActiveSheet.Cells(3, 3) = "Entropy (S)"
  ActiveSheet.Cells(3, 4) = "Lambda"
  ActiveSheet.Cells(3, 5) = "Entropy(Lambda_k)"
  ActiveSheet.Cells(3, 6) = "delta lambda"
  ActiveSheet.Cells(3, 7) = "Lambda from beta goal"
  ActiveSheet.Cells(3, 8) = "Beta goal to min R^2"
  ActiveSheet.Cells(3, 9) = "Mindshare"
  ActiveSheet.Cells(2, 9) = "beta / authors"
'get formula of trendlineEq of entropy power trend from the entropy lambda sheet
    trendEq_entropy = Sheets(sourceSheet).Cells(1, 7)
    m_s = firstPartTrendEq(trendEq_entropy)
    b_s = secondPartTrendEq(trendEq_entropy)

    Sheets(currentName).Cells(1, 2) = b_s
    Sheets(currentName).Cells(2, 2) = m_s

'get formula of trendlineEq of the messages graph from the entropy lamda sheet
    trendEq_lambda = Sheets(sourceSheet).Cells(1, 8)
    m_lambda = firstPartTrendEq(trendEq_lambda)
    b_lambda = secondPartTrendEq(trendEq_lambda)

    Sheets(currentName).Cells(1, 4) = b_lambda
    Sheets(currentName).Cells(2, 4) = m_lambda

Sheets(currentName).Select

'fill in columns
  For i = startRange + 1 To numRows
    Sheets(currentName).Cells(i, 1) = Sheets(sourceSheet).Cells(i, 1)
    Sheets(currentName).Cells(i, 2) = Sheets(sourceSheet).Cells(i, 2)
    Sheets(currentName).Cells(i, 3) = Sheets(sourceSheet).Cells(i, 4)
    Sheets(currentName).Cells(i, 4) = Sheets(sourceSheet).Cells(i, 9)
    Sheets(currentName).Cells(i, 5) = "=$B$2*(G" & i & "/$D2)^(B1/D1)"
       '          G              m_s
       ' b_s * (-------- ) ^ ( ------------- )
       '       bLambda          m_lambda


    Sheets(currentName).Cells(i, 6) = "=(C" & i & "-E" & i & ")^2"
    Sheets(currentName).Cells(i, 7) = "=((H" & i & "* EntropyLambda!F" & i & ")/((H" _
       & i & "* EntropyLambda!F" & i & ") + EntropyLambda!G" & i & "))^(1/3)"
    '     ((H*F)
    '  ------------
    '   ((H*F)+G))       ^(1/3)


    Sheets(currentName).Cells(i, 8) = betaPercentage
    'goal seek the difference between actual entropy and calculated.. so delta
    'approaches zero.
    Range("F" & i).GoalSeek Goal:=0, ChangingCell:=Range("H" & i)
    If Sheets(currentName).Cells(i, 8) > 1# Or Sheets(currentName).Cells(i, 8) < 0# Then
       Sheets(currentName).Cells(i, 8) = betaPercentage
```

```
      Range("F" & i).GoalSeek Goal:=0, ChangingCell:=Range("H" & i)
    End If

    If Sheets(currentName).Cells(i, 8) > 1# Or Sheets(currentName).Cells(i, 8) < 0# Then
       Sheets(currentName).Cells(i, 8) = betaPercentage
    End If

    'mindshare = beta / authors
    Sheets(currentName).Cells(i, 9) = "=(H" & i & "/" & sourceSheet2 & "!D" & i & ")"
  Next i

Call CopyCalc_beta_EntropyGraph(startRange, numRows, currentName)
Call CopyMindshareGraph(startRange, numRows, currentName)

End Sub 'correlate_s_e

'--------------------------------------------------------------------------
'  Sub: CopyMessagesGraph
'  Author: Matt Behnke
'  Created: 2/11/02
'  Description: copies the graph of messages (Records, N)
'  inputs: start range, num of rows on the source sheet
'
'  Outputs:
'--------------------------------------------------------------------------
Sub CopyMessagesGraph(ByVal startRange As Integer, ByVal endRange As Integer, ByVal source As
String)

    sheetCount = Sheets.Count
    Application.DisplayAlerts = False

'GRAPH "Messages"

    Windows("EntropyLambdaMacro.xls").Activate
    Sheets("Messages").Select
    Sheets("Messages").Copy After:=Workbooks(currFilename).Sheets(sheetCount)

    ActiveChart.SeriesCollection(1).Select

    'msgs - Records, N
    ActiveChart.SeriesCollection(1).Values = "=" & source & "!R" & startRange & "C3:R" & endRange &
"C3"
    'timesteps
    ActiveChart.SeriesCollection(1).XValues = "=" & source & "!R" & startRange & "C1:R" & endRange &
"C1"

    With ActiveChart.SeriesCollection(1).Trendlines(1)
    'put trendline equation onto source
    .DisplayEquation = True
    .DisplayRSquared = True
     Worksheets(source).Cells(1, 6).Value = .DataLabel.Text
   End With

End Sub 'copy messages graph

'--------------------------------------------------------------------------
```

```
'  Sub: CopyMapEntropyKK_1
'  Author: Matt Behnke
'  Created: 2/13/02
'  Description: copies the graph of the map of entropy of timestep k& k+1
'  inputs: start range, num of rows on the source sheet
'
'  Outputs:
'--------------------------------------------------------------------------
Sub CopyMapEntropyKK_1(ByVal startRange As Integer, ByVal endRange As Integer, ByVal source As
String)

    sheetCount = Sheets.Count

    Application.DisplayAlerts = False

'GRAPH "Messages"

    Windows("EntropyLambdaMacro.xls").Activate
    Sheets("Map Entropy K, K+1").Select
    Sheets("Map Entropy K, K+1").Copy After:=Workbooks(currFilename).Sheets(sheetCount)

    ActiveChart.SeriesCollection(1).Select

    'entropy k,k+1
    ActiveChart.SeriesCollection(1).Values = "=" & source & "!R" & startRange & "C5:R" & endRange - 1
& "C5"
    'entropy k
    ActiveChart.SeriesCollection(1).XValues = "=" & source & "!R" & startRange & "C4:R" & endRange -
1 & "C4"

    With ActiveChart.SeriesCollection(1).Trendlines(1)
    'put trendline equation onto source
    .DisplayEquation = True
    .DisplayRSquared = True
     'Worksheets(source).Cells(1, 6).Value = .DataLabel.Text this trendline eq is not used.
   End With

End Sub 'copy map k k+1 graph

'--------------------------------------------------------------------------
'  Sub: CopyEntropyLambdaChart
'  Author: Matt Behnke
'  Created: 2/14/02
'  Description: copies the graph of the entropy lambda chart
'  inputs: start range, num of rows on the source sheet
'          update - if true it updates the trendline for series 1..
'              this is because the graph is added to the worksheet
'          before the lambda data is there.
'  Outputs:
'--------------------------------------------------------------------------
Sub CopyEntropyLambdaChart(ByVal startRange As Integer, ByVal endRange As Integer, ByVal source
As String, ByVal update As Boolean)

If update = False Then
    sheetCount = Sheets.Count
```

```
    Application.DisplayAlerts = False

'GRAPH "Messages"

    Windows("EntropyLambdaMacro.xls").Activate
    Sheets("Entropy Lambda Chart").Select
    Sheets("Entropy Lambda Chart").Copy After:=Workbooks(currFilename).Sheets(sheetCount)

    ActiveChart.SeriesCollection(1).Select

    'lambda
    ActiveChart.SeriesCollection(1).Values = "=" & source & "!R" & startRange & "C9:R" & endRange &
"C9"
    'entropy S(k)
    ActiveChart.SeriesCollection(2).Values = "=" & source & "!R" & startRange & "C4:R" & endRange &
"C4"

    With ActiveChart.SeriesCollection(2).Trendlines(1)
    'put trendline equation onto source
    .DisplayEquation = True
    .DisplayRSquared = True
    Worksheets(source).Cells(1, 7).Value = .DataLabel.Text 'this trendline eq is not used.
   End With

Else ' If update = True Then
        With ActiveChart.SeriesCollection(1).Trendlines(1)
        'put trendline equation onto source
        .DisplayEquation = True
        .DisplayRSquared = True
        Worksheets(source).Cells(1, 8).Value = .DataLabel.Text
        End With
End If 'update

End Sub 'Copy Entropy Lambda Chart


'--------------------------------------------------------------------------
'  Sub: CopyCalc_beta_EntropyGraph
'  Author: Matt Behnke
'  Created: 2/14/02
'  Description: copies the graph of the entropy lambda calculated chart
'  inputs: start range, num of rows on the source sheet
'
'          before the lambda data is there.
'  Outputs:
'--------------------------------------------------------------------------
Sub CopyCalc_beta_EntropyGraph(ByVal startRange As Integer, ByVal endRange As Integer, ByVal
source As String)

    sheetCount = Sheets.Count

    Application.DisplayAlerts = False

    Windows("EntropyLambdaMacro.xls").Activate
    Sheets("Calc_beta_Entropy (lambda) Chrt").Select
    Sheets("Calc_beta_Entropy (lambda) Chrt").Copy After:=Workbooks(currFilename).Sheets(sheetCount)
```

```
    ActiveChart.SeriesCollection(1).Select

    'entropy S, actual
    ActiveChart.SeriesCollection(1).Values = "=" & source & "!R" & startRange & "C3:R" & endRange &
"C3"
    ActiveChart.SeriesCollection(1).XValues = "=" & source & "!R" & startRange & "C1:R" & endRange &
"C1"
    'entropy S as a function of lambda
    ActiveChart.SeriesCollection(2).Values = "=" & source & "!R" & startRange & "C5:R" & endRange &
"C5"
    ActiveChart.SeriesCollection(1).XValues = "=" & source & "!R" & startRange & "C1:R" & endRange &
"C1"
    'beta_k_s(actual) .. from the goal seek to minize the diff between the above two.
    ActiveChart.SeriesCollection(3).Values = "=" & source & "!R" & startRange & "C8:R" & endRange &
"C8"
    ActiveChart.SeriesCollection(1).XValues = "=" & source & "!R" & startRange & "C1:R" & endRange &
"C1"

End Sub 'Copy Entropy Lambda Chart

'--------------------------------------------------------------------------
'  Sub: CopyCalc_beta_EntropyGraph
'  Author: Matt Behnke
'  Created: 5/7/02
'  Description: copies the graph of mindshare
'  inputs: start range, num of rows on the source sheet
'  Outputs:
'--------------------------------------------------------------------------
Sub CopyMindshareGraph(ByVal startRange As Integer, ByVal endRange As Integer, ByVal source As
String)

    Dim sourceSheet2 As String                  'holds the name of the affiliation summary sheet
    sourceSheet2 = "Affiliation_Summary"

    sheetCount = Sheets.Count

    Application.DisplayAlerts = False

    Windows("EntropyLambdaMacro.xls").Activate
    Sheets("Mindshare").Select
    Sheets("Mindshare").Copy After:=Workbooks(currFilename).Sheets(sheetCount)

    ActiveChart.SeriesCollection(1).Select

    'author instances
    ActiveChart.SeriesCollection(2).Values = "=" & sourceSheet2 & "!R" & startRange & "C4:R" &
endRange & "C4"
    ActiveChart.SeriesCollection(1).XValues = "=" & source & "!R" & startRange & "C1:R" & endRange &
"C1"
    'beta_k per author ..
    ActiveChart.SeriesCollection(1).Values = "=" & source & "!R" & startRange & "C9:R" & endRange &
"C9"
    ActiveChart.SeriesCollection(1).XValues = "=" & sourceSheet2 & "!R" & startRange & "C1:R" &
endRange & "C1"
```

```
End Sub 'Copy mindshare graph
'--------------------------------------------------------------------------
'  Function: rSquaredTrendEq
'  Author: Matt Behnke
'  Created: 1/3/02
'  Description: extracts the rSquared value of a trendline equation
'  inputs:   trendline equation
'  Outputs: firstpart of trendline equation
'--------------------------------------------------------------------------
Function rSquaredTrendEq(ByVal trendlineEq As String) As Double

    tempStorage = Sheets(1).Cells(1, 1)
    Sheets(1).Cells(1, 1) = trendlineEq

    i = 1
    While found = False
       testchar = Sheets(1).Cells(1, 1).Characters(i, 1).Text
       If testchar = "R" Then
          found = True
       Else
       i = i + 1
       End If
     Wend

    'i = location of R
    'secondpart starts at character i plus 5..
    'num of characters = location(x) - 5
    'extract 5 characters..

    rSquaredTrendEq = Sheets(1).Cells(1, 1).Characters(i + 5, 6).Text
    Sheets(1).Cells(1, 1) = tempStorage

End Function ' rSquaredTrendEqu

'--------------------------------------------------------------------------
'  Function: CountRows
'  Author: ? Revised by: Matt Behnke
'  Created: ?
'  Revised: 9/10/01
'  Description: Counts the rows in the suppiled worksheet and column number
'  inputs:   sheetName - name of the sheet to count the rows in
'            colNum - number of the column to count rows in
'  Outputs: number of rows as a double
'--------------------------------------------------------------------------
Function CountRows(ByVal sheetname As String, ByVal colNum As Integer) As Double
On Error Resume Next
Dim currCell As Range, rowNum As Double

  Sheets("" & sheetname).Select

  If IsNumeric(colNum) Then
  Else
    colNum = 1
  End If

  rowNum = 1
```

```
  Set currCell = ActiveSheet.Cells(rowNum, colNum)
  Do While currCell.Value <> ""
    rowNum = rowNum + 1
    Set currCell = ActiveSheet.Cells(rowNum, colNum)
  Loop
  CountRows = rowNum - 1
End Function 'CountRows
'--------------------------------------------------------------------------
'  Function: CountCols
'  Author: ? Revised by: Matt Behnke
'  Created: ?
'  Revised: 9/10/01
'  Description: Counts the rows in the suppiled worksheet and column number
'  inputs:   sheetName - name of the sheet to count the columns in
'            rowNum - number of the row to count columns in
'  Outputs: number of columns as a double
'--------------------------------------------------------------------------
Function CountCols(ByVal sheetname As String, ByVal rowNum As Integer) As Integer
On Error Resume Next
Dim currCell As Range, colNum As Integer

  Sheets("" & sheetname).Select

  If IsNumeric(rowNum) Then
  Else
    rowNum = 1
  End If
  colNum = 1
  Set currCell = ActiveSheet.Cells(rowNum, colNum)
  Do While currCell.Value <> ""
    colNum = colNum + 1
    Set currCell = ActiveSheet.Cells(rowNum, colNum)
  Loop
  CountCols = colNum - 1
End Function 'CountCols
'--------------------------------------------------------------------------
'  Function: firstPartTrendEq
'  Author: Matt Behnke
'  Created: 11/13/01
'  Description: extracts the first part of the given trendline equation
'  inputs:   trendline equation
'  Outputs: firstpart of trendline equation
'--------------------------------------------------------------------------
Function firstPartTrendEq(ByVal trendlineEq As String) As Double

    tempStorage = Sheets(1).Cells(1, 1)
    Sheets(1).Cells(1, 1) = trendlineEq

    i = 1
    While found = False
        testchar = Sheets(1).Cells(1, 1).Characters(i, 1).Text
        If testchar = "x" Then
            found = True
        Else
        i = i + 1
        End If
```

```
    Wend

   'i = location of x
   'firstpart = starts at character 5
   'num of characters = location(x) - 5

   firstPartTrendEq = Sheets(1).Cells(1, 1).Characters(5, i - 5).Text
   Sheets(1).Cells(1, 1) = tempStorage

End Function ' first part trendline
'--------------------------------------------------------------------------
'  Function: secondPartTrendEq
'  Author: Matt Behnke
'  Created: 11/13/01
'  Description: extracts the first part of the given trendline equation
'  inputs:   trendline equation
'  Outputs: firstpart of trendline equation
'--------------------------------------------------------------------------
Function secondPartTrendEq(ByVal trendlineEq As String) As Double

   tempStorage = Sheets(1).Cells(1, 1)
   Sheets(1).Cells(1, 1) = trendlineEq

   i = 1
   While found = False
      testchar = Sheets(1).Cells(1, 1).Characters(i, 1).Text
      If testchar = "x" Then
         found = True
      Else
      i = i + 1
      End If
    Wend

   'i = location of x
   'secondpart starts at character i plus 1..
   'num of characters = location(x) - 5
   'extract 5 characters..

   j = i
   While found2 = False
      testchar = Sheets(1).Cells(1, 1).Characters(j, 1).Text
      If testchar = "R" Then
         found2 = True
      Else
      j = j + 1
      End If
    Wend
   stopchar = (j - 1) - (i + 1)

   secondPartTrendEq = Sheets(1).Cells(1, 1).Characters(i + 1, stopchar).Text
   Sheets(1).Cells(1, 1) = tempStorage

End Function ' secondPart eq
```

APPENDIX P – Q LEVEL ANALYSIS MACRO CODE

```
'--------------------------------------------------------------------------
'  Macro: Q Level Analysis
'  Author: Matt Behnke
'  Created: 1/18/02
'  Description: Contains the qlevel analysis functions – code is needed from
'                  the affiliation distribution macro. Will need to separate this dependency.
'  inputs:
'  Outputs:
'--------------------------------------------------------------------------


'--------------------------------------------------------------------------
'  sub: qLevelTrigger
'  Author: Matt Behnke
'  Created: 1/18/02
'  Description: activates the qlevel functions – some code maybe needed from
'                  the affiliation distribution macro
'  inputs:
'  Outputs:
'--------------------------------------------------------------------------
Sub qLevelTrigger()

Dim graphStart As Integer

qLevelType = InputBox("Enter Author or Term:")
numQLevels = InputBox("Enter number of qLevels:")

If qLevelType = "Author" Or qLevelType = "Term" And numQLevels > 0 Then 'check input

   If qLevelType = "Author" Then
      prefix = "_"
   Else
      prefix = ""
   End If

   Call qLevelCumulative(qLevelType, prefix, numQLevels)
   Call qLevelSummary(qLevelType, prefix, numQLevels)
   Call qLevelYears(qLevelType, prefix, numQLevels)
    Call qLevelCopyYearsGraphs(numQLevels)
   Call qLevelMonths(qLevelType, prefix, numQLevels, False)
   Call qLevelMonths(qLevelType, prefix, numQLevels, True) 'calculate mass
   Call qLevelMonthsCount(qLevelType, prefix, numQLevels)
   Call qLevelEntropy(qLevelType, prefix, numQLevels)
   Call qLevelMonthTemp(numQLevels)
   Call qLevelEntropy2(qLevelType, prefix, numQLevels)

   If qLevelType = "Term" Then
      graphStart = qLevelGraphStart("q_level_monthly_temp", 3, 4)
      Call qLevelCopyGraphs(graphStart, numQLevels)
   End If

Else
```

```
   MsgBox ("WRONG INPUT.. TRY AGAIN!")
End If

End Sub

'--------------------------------------------------------------------------
'  sub: qLevelCumulative
'  Author: Matt Behnke
'  Created: 1/16/02
'  Description: ..puts in cumulative values for each time step, and sums at the bottom
'  inputs:  qlevelType: author or term
'           prefix: the sheet prefix, changes whether its author or term
'           numqLevels: the number of q levels
'  Outputs:
'--------------------------------------------------------------------------
Sub qLevelCumulative(ByVal qLevelType As String, ByVal prefix As String, ByVal numQLevels As
Integer)

For i = 1 To numQLevels

   If i < 10 Then
      Call CalcCumulative(prefix & "0" & i & "_" & qLevelType & "_month")
   Else
      Call CalcCumulative(prefix & "" & i & "_" & qLevelType & "_month")
   End If
Next i

End Sub

'--------------------------------------------------------------------------
'  sub: qLevelSummary
'  Author: Matt Behnke
'  Created: 1/14/02
'  Description: creates a summary sheet for a q_level, lists time steps and num of instances
'               for each q level
'  inputs:  qlevelType: author or term
'           prefix: the sheet prefix, changes whether its author or term
'           numqLevels: the number of q levels
'  Outputs:
'--------------------------------------------------------------------------
Sub qLevelSummary(ByVal qLevelType As String, ByVal prefix As String, ByVal numQLevels As
Integer)

For z = 1 To numQLevels
   If z < 10 Then
      sheetName = prefix & "0" & z & "_" & qLevelType & "_month"
      summarySheet = prefix & "0" & z & "_" & qLevelType & "_summary"
   Else
      sheetName = prefix & "" & z & "_" & qLevelType & "_month"
      summarySheet = prefix & "" & z & "_" & qLevelType & "_summary"
   End If

   Sheets.Add After:=Sheets(Sheets.Count)
   'Sheets(Sheets.Count).Select
   ActiveSheet.Name = summarySheet
```

```
   numColumns = CountCols(sheetName, 1)
   numRows = CountRows(sheetName, 1)

   Sheets(summarySheet).Cells(1, 1) = "q level"
   Sheets(summarySheet).Cells(1, 2) = z


   Sheets(summarySheet).Cells(2, 1) = " "
   Sheets(summarySheet).Cells(3, 1) = "Time Steps"
   Sheets(summarySheet).Cells(3, 2) = "sum"
   Sheets(summarySheet).Cells(3, 3) = "count"

   For i = 4 To numColumns
      Sheets(summarySheet).Cells(i, 1) = Sheets(sheetName).Cells(1, i)
      Sheets(summarySheet).Cells(i, 2) = "=SUM('" & sheetName & "'!" & col(i) & "2:" & col(i) &
numRows & ")"
      Sheets(summarySheet).Cells(i, 3) = "=Count('" & sheetName & "'!" & col(i) & "2:" & col(i) &
numRows & ")"
   Next i
Next z

End Sub


'---------------------------------------------------------------------------
'  sub: qLevelYears
'  Author: Matt Behnke
'  Created: 1/16/02
'  Description: uses an array of years to store the amount of instances for a year.
'               outputs each year of the dataset to each summary sheet and number of instances
'  inputs:  qlevelType: author or term
'           prefix: the sheet prefix, changes whether its author or term
'           numqLevels: the number of q levels
'  Outputs:
'---------------------------------------------------------------------------
Sub qLevelYears(ByVal qLevelType As String, ByVal prefix As String, ByVal numQLevels As Integer)

'number of ntuples
nTuples = numQLevels

Dim years As Variant

overallQSummary = "q_summary_year"

firstYear = "2500"
firstYearOffset = 6
lastYear = 0

Sheets.Add After:=Sheets(Sheets.Count)
'Sheets(Sheets.Count).Select
ActiveSheet.Name = overallQSummary

Cells(1, 1) = " "
Cells(2, 1) = " "
Cells(3, 1) = " "
```

```
For z = 1 To nTuples

   years = Array(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)

   If z < 10 Then
      summarySheet = prefix & "0" & z & "_" & qLevelType & "_summary"
   Else
      summarySheet = prefix & "" & z & "_" & qLevelType & "_summary"
   End If

   numRows = CountRows(summarySheet, 1)

   For i = 4 To numRows

      'determine current year
      j = 1
      While found = False
         testchar = Cells(i, 1).Characters(j, 1).Text
         If testchar = "/" Then
            found = True
         Else
         j = j + 1
         End If
      Wend
      'month ends at j

      currentMonth = Cells(i, 1).Characters(1, j - 1).Text
      If currentMonth < 10 Then
         theYear = Cells(i, 1).Characters(5, 4).Text
      Else
         theYear = Cells(i, 1).Characters(6, 4).Text
      End If

      'If theYear > 50 Then 'add prefix to the year
      '    theYear = "19" & theYear
      'Else
      '    theYear = "20" & theYear
      'End If

      'check to see if the currentYear is less than first year
      If theYear < firstYear Then
         firstYear = theYear
         firstYearOffset = firstYearOffset - 1 'array index
      End If

      If theYear > lastYear Then
         lastYear = theYear
      End If

      yearOffset = theYear - firstYear + 5  'so there can be 5 years less data than the first firstYear value
      years(yearOffset) = Sheets(summarySheet).Cells(i, 2)
      found = False
   Next i

   'output yearsArray
```

```
    Sheets(summarySheet).Cells(3, 4) = "Years"
    Sheets(summarySheet).Cells(3, 5) = "instances"

    counter = 4 'for output


    For x = firstYearOffset To lastYear - firstYear + firstYearOffset
        Sheets(summarySheet).Cells(counter, 4) = firstYear + x - firstYearOffset
        Sheets(summarySheet).Cells(counter, 5) = years(x) 'instances
        If z = 1 Then
            Sheets(overallQSummary).Cells(counter, z) = firstYear + x - firstYearOffset
            Sheets(overallQSummary).Cells(counter, z + 1) = 0
        End If
        Sheets(overallQSummary).Cells(counter, z + 2) = years(x)

        'if there is a zero in a year then the put the previous years value into the current year (cumulative)
        If Sheets(overallQSummary).Cells(counter, z + 2) = 0 And x > firstYearOffset Then
            Sheets(overallQSummary).Cells(counter, z + 2) = Sheets(overallQSummary).Cells(counter - 1, z +
2)
        End If

        If x = firstYearOffset Then
            Sheets(overallQSummary).Cells(counter - 1, z + 2) = z
        End If


    counter = counter + 1

    Next x

Next z
'copy chart

    'currFilename = Application.ActiveWorkbook.Name
    'Windows("AffiliationMacro.xls").Activate
    'Sheets("q_level_yr").Select
    'Sheets("q_level_yr").Copy After:=Workbooks(currFilename).Sheets(1)
    '
    'counter = 4
    'columnStart = 2
    'For z = 2 To nTuples
        'ActiveChart.SeriesCollection(z - 1).Values = "=" & overallQSummary & "!R" & counter & "C" &
columnStart & ":R" & counter & "C" & ntuples
    '    counter = counter + 1
    'Next z

End Sub

'---------------------------------------------------------------------------
'  sub: qLevelMonths
'  Author: Matt Behnke
'  Created: 1/27/02 --------finished 2/4/02
'  Description: uses an array of years and months to store the amount of instances for each timestep k.
'               outputs the number of cumulative instances per month / year
'  inputs:  qlevelType: author or term
'           prefix: the sheet prefix, changes whether its author or term
```

```
'          numqLevels: the number of q levels
'  Outputs:
'--------------------------------------------------------------------------
Sub qLevelMonths(ByVal qLevelType As String, ByVal prefix As String, ByVal numQLevels As Integer,
ByVal mass As Boolean)

'number of ntuples
nTuples = numQLevels

Dim years As Variant
Dim yearsMonths As Variant

firstYear = "2500"
lastYear = 0

If mass = True Then
   overallQSummary = "q_summary_monthly_count_mass"
Else
   overallQSummary = "q_summary_monthly_instances"
End If

Sheets.Add After:=Sheets(Sheets.Count)
'Sheets(Sheets.Count).Select
ActiveSheet.Name = overallQSummary

Sheets(overallQSummary).Cells(1, 1) = "  "
Sheets(overallQSummary).Cells(2, 1) = "  "
Sheets(overallQSummary).Cells(3, 1) = "  "

Cells(4, 2).Select
ActiveWindow.FreezePanes = True

years = Array()
yearsMonths = Array()
'ReDim yearsMonths(0 To 0, 1 To 12)

For z = 1 To nTuples

   'get the source sheet's name
   If z < 10 Then
      summarySheet = prefix & "0" & z & "_" & qLevelType & "_summary"
   Else
      summarySheet = prefix & "" & z & "_" & qLevelType & "_summary"
   End If

   numRows = CountRows(summarySheet, 1)

   'scan the first sheet to get the last and first year to get the array bounds
   If z = 1 Then

      For i = 4 To numRows

         'determine current year
         j = 1
         While found = False
            testchar = Cells(i, 1).Characters(j, 1).Text
```

```
            If testchar = "/" Then
               found = True
            Else
            j = j + 1
            End If
         Wend
         'month ends at j

         currentMonth = Cells(i, 1).Characters(1, j - 1).Text
         If currentMonth < 10 Then
            theYear = Cells(i, 1).Characters(5, 4).Text
         Else
            theYear = Cells(i, 1).Characters(6, 4).Text
         End If

         'If theYear > 50 Then 'add prefix to the year
         '    theYear = "19" & theYear
         'Else
         '    theYear = "20" & theYear
         'End If

         'check to see if the currentYear is less than first year
         If theYear < firstYear Then
            firstYear = theYear
         End If

         If theYear > lastYear Then
            lastYear = theYear
         End If

         found = False
      Next i 'done scanning the sheet now redim the array

      'ReDim yearsMonths(firstYear To lastYear, 1 To 12)

   End If 'z = 1

   ReDim yearsMonths(firstYear To lastYear, 1 To 12)

   For i = 4 To numRows 'now process all the nTuple sheets

      'determine current year
      j = 1
      While found = False
         testchar = Cells(i, 1).Characters(j, 1).Text
         If testchar = "/" Then
            found = True
         Else
         j = j + 1
         End If
      Wend
      'month ends at j

      currentMonth = Cells(i, 1).Characters(1, j - 1).Text
      If currentMonth < 10 Then
         theYear = Cells(i, 1).Characters(5, 4).Text
```

```
    Else
        theYear = Cells(i, 1).Characters(6, 4).Text
    End If

    'If theYear > 50 Then 'add prefix to the year
    '    theYear = "19" & theYear
    'Else
    '    theYear = "20" & theYear
    'End If

    yearsMonths(theYear, currentMonth) = Sheets(summarySheet).Cells(i, 2)
    found = False
Next i

'output yearsArray

'Sheets(summarySheet).Cells(3, 4) = "Years"
'Sheets(summarySheet).Cells(3, 5) = "instances"

counter = 4 'for output
monthCounter = 1
lastValue = 0


For j = firstYear To lastYear

    For k = 1 To 12

        If "" & j = firstYear And k = 1 Then 'label the q levels
            Sheets(overallQSummary).Cells(counter - 1, z + 2) = z
        End If

        If z = 1 Then 'put the month/year
            Sheets(overallQSummary).Cells(counter, z) = "" & monthCounter & "/" & j
            Sheets(overallQSummary).Cells(counter, z + 1) = 0
        End If

        currentValue = yearsMonths(j, monthCounter)
        If mass = True Then
            multiplyer = z
        Else
            multiplyer = 1
        End If

        If currentValue > lastValue Then
            Sheets(overallQSummary).Cells(counter, z + 2) = currentValue * multiplyer
            lastValue = currentValue
        Else
            Sheets(overallQSummary).Cells(counter, z + 2) = lastValue * multiplyer
        End If

        monthCounter = monthCounter + 1
        If monthCounter > 12 Then
            monthCounter = 1
        End If
```

```
        If z = nTuples Then
           Sheets(overallQSummary).Cells(counter, z + 3) = "=SUM(" & col(2) & counter _
                       & ":" & col(nTuples + 2) & counter & ")"
        End If

        counter = counter + 1

      Next k
   Next j
   'If z = nTuples Then
   '    Sheets(overallQSummary).Cells(counter - 1, z + 3) = "=SUM(" * col(2) & counter - 1 _
                       & ":" & col(nTuples + 2) & counter - 1 & ")"
  ' End If


Next z
'total of all the qlevels:
Sheets(overallQSummary).Cells(1, 1) = "=SUM(" & col(2) & counter - 1 & ":" & col(nTuples + 2) &
counter - 1 & ")"



'copy chart

'    currFilename = Application.ActiveWorkbook.Name

 '   Windows("AffiliationMacro.xls").Activate
  '  Sheets("q_level_yr").Select
   ' Sheets("q_level_yr").Copy After:=Workbooks(currFilename).Sheets(Sheets.Count)

   ' counter = 4
   ' columnStart = 2
   ' For z = 2 To nTuples
   '     ActiveChart.SeriesCollection(z - 1).Values = "=" & overallQSummary & "!R" & counter & "C" &
columnStart & ":R" & counter & "C" & nTuples
   '     counter = counter + 1
   ' Next z

End Sub 'qlevel months

'--------------------------------------------------------------------------
'  sub: qLevelMonthsCOUNT
'  Author: Matt Behnke
'  Created: 2/17/02
'  Description: uses an array of years and months to store the amount of instances for each timestep k.
'               outputs the number of terms in the vocab per month / year puts
'  inputs:  qlevelType: author or term
'           prefix: the sheet prefix, changes whether its author or term
'           numqLevels: the number of q levels

'  Outputs:
'--------------------------------------------------------------------------
Sub qLevelMonthsCount(ByVal qLevelType As String, ByVal prefix As String, ByVal numQLevels As
Integer)

'number of ntuples
```

```
nTuples = numQLevels

Dim years As Variant
Dim yearsMonths As Variant

firstYear = "2500"
lastYear = 0

   overallQSummary = "q_summary_monthly_count_count"

Sheets.Add After:=Sheets(Sheets.Count)
'Sheets(Sheets.Count).Select
ActiveSheet.Name = overallQSummary

Sheets(overallQSummary).Cells(1, 1) = "  "
Sheets(overallQSummary).Cells(2, 1) = "  "
Sheets(overallQSummary).Cells(3, 1) = "  "

Cells(4, 2).Select
ActiveWindow.FreezePanes = True

years = Array()
yearsMonths = Array()
'ReDim yearsMonths(0 To 0, 1 To 12)

For z = 1 To nTuples

   'get the source sheet's name
   If z < 10 Then
      summarySheet = prefix & "0" & z & "_" & qLevelType & "_summary"
   Else
      summarySheet = prefix & "" & z & "_" & qLevelType & "_summary"
   End If

   numRows = CountRows(summarySheet, 1)

   'scan the first sheet to get the last and first year to get the array bounds
   If z = 1 Then

      For i = 4 To numRows

         'determine current year
         j = 1
         While found = False
            testchar = Cells(i, 1).Characters(j, 1).Text
            If testchar = "/" Then
               found = True
            Else
            j = j + 1
            End If
         Wend
         'month ends at j

         currentMonth = Cells(i, 1).Characters(1, j - 1).Text
         If currentMonth < 10 Then
            theYear = Cells(i, 1).Characters(5, 4).Text
```

```
        Else
           theYear = Cells(i, 1).Characters(6, 4).Text
        End If

        'If theYear > 50 Then 'add prefix to the year
        '    theYear = "19" & theYear
        'Else
        '    theYear = "20" & theYear
        'End If

        'check to see if the currentYear is less than first year
        If theYear < firstYear Then
           firstYear = theYear
        End If

        If theYear > lastYear Then
           lastYear = theYear
        End If

        found = False
    Next i 'done scanning the sheet now redim the array

    'ReDim yearsMonths(firstYear To lastYear, 1 To 12)

End If 'z = 1

ReDim yearsMonths(firstYear To lastYear, 1 To 12)

For i = 4 To numRows 'now process all the nTuple sheets

    'determine current year
    j = 1
    While found = False
       testchar = Cells(i, 1).Characters(j, 1).Text
       If testchar = "/" Then
          found = True
       Else
       j = j + 1
       End If
    Wend
    'month ends at j

    currentMonth = Cells(i, 1).Characters(1, j - 1).Text
    If currentMonth < 10 Then
       theYear = Cells(i, 1).Characters(5, 4).Text
    Else
       theYear = Cells(i, 1).Characters(6, 4).Text
    End If

    'If theYear > 50 Then 'add prefix to the year
    '    theYear = "19" & theYear
    'Else
    '    theYear = "20" & theYear
    'End If

    yearsMonths(theYear, currentMonth) = Sheets(summarySheet).Cells(i, 3) 'count
```

```
      found = False
   Next i

   counter = 4 'for output
   monthCounter = 1
   lastValue = 0
   'numYears = UBound(yearsMonthsa)
'*****************************************************WORK ON THIS later!!

   For j = firstYear To lastYear

      For k = 1 To 12

         If "" & j = firstYear And k = 1 Then 'label the q levels
            Sheets(overallQSummary).Cells(counter - 1, z + 2) = z
         End If

         If z = 1 Then 'put the month/year
            Sheets(overallQSummary).Cells(counter, z) = "'" & monthCounter & "/" & j
            Sheets(overallQSummary).Cells(counter, z + 1) = 0
         End If

         currentValue = yearsMonths(j, monthCounter)
         If mass = True Then
            multiplyer = z
         Else
            multiplyer = 1
         End If

         If currentValue > lastValue Then
            Sheets(overallQSummary).Cells(counter, z + 2) = currentValue * multiplyer
            lastValue = currentValue
         Else
            Sheets(overallQSummary).Cells(counter, z + 2) = lastValue * multiplyer
         End If

         monthCounter = monthCounter + 1
         If monthCounter > 12 Then
            monthCounter = 1
         End If

         If z = nTuples Then
            Sheets(overallQSummary).Cells(counter, z + 3) = "=SUM(" & col(2) & counter _
                        & ":" & col(nTuples + 2) & counter & ")"
         End If


         counter = counter + 1

      Next k
   Next j
'    If z = nTuples Then
'       Sheets(overallQSummary).Cells(counter - 1, z + 3) = "=SUM(" * col(2) & counter - 1 _
'                           & ":" & col(nTuples + 2) & counter - 1 & ")"
'    End If
```

```
Next z
'total of all the qlevels:
Sheets(overallQSummary).Cells(1, 1) = "=SUM(" & col(2) & counter - 1 & ":" & col(nTuples + 2) &
counter - 1 & ")"

End Sub 'qlevel months CoUNT



'--------------------------------------------------------------------------
'  sub: qLevelEntropy
'  Author: Matt Behnke
'  Created:  2/4/02
'  finished: 3/11/02
'  Description: uses an array of years and months to store the amount of entropy for each timestep k.
'               outputs the cumulative entropy per timestep per q level
'  inputs:  qlevelType: author or term
'           prefix: the sheet prefix, changes whether its author or term
'           numqLevels: the number of q levels
'  Outputs:
'--------------------------------------------------------------------------
Sub qLevelEntropy(ByVal qLevelType As String, ByVal prefix As String, ByVal numQLevels As Integer)

'number of ntuples
nTuples = numQLevels

Dim years As Variant
Dim yearsMonths As Variant
Dim contributionEntropy As Variant

firstYear = "2500"
lastYear = 0

overallQEntropy = "q_local_entropy_monthly"
contributionEntropySheet = "q_contribution_entropy_monthly"
countSheet = "q_summary_monthly_instances"

'add the contribution entropy sheet
Sheets.Add After:=Sheets(Sheets.Count)
'Sheets(Sheets.Count).Select
ActiveSheet.Name = contributionEntropySheet
Cells(4, 2).Select
ActiveWindow.FreezePanes = True

'add the local entropy sheet
Sheets.Add After:=Sheets(Sheets.Count)
'Sheets(Sheets.Count).Select
ActiveSheet.Name = overallQEntropy
Cells(4, 2).Select
ActiveWindow.FreezePanes = True

years = Array()
yearsMonths = Array()
contributionEntropy = Array()
```

```
'ReDim yearsMonths(0 To 0, 1 To 12)

For z = 1 To nTuples

   'get the source sheet's name
   If z < 10 Then
      summarySheet = prefix & "0" & z & "_" & qLevelType & "_month"
   Else
      summarySheet = prefix & "" & z & "_" & qLevelType & "_month"
   End If

   numRows = CountRows(summarySheet, 1)
   numCols = CountCols(summarySheet, 1)

   'find the row that contains the time counts for the current time step...

   'scan the first sheet to get the last and first year to get the array bounds
   If z = 1 Then

      For i = 4 To numCols

         'determine current year
         j = 1
         While found = False
            testchar = Cells(1, i).Characters(j, 1).Text
            If testchar = "/" Then
               found = True
            Else
            j = j + 1
            End If
         Wend
         'month ends at j

         currentMonth = Cells(1, i).Characters(1, j - 1).Text
         If currentMonth < 10 Then
            theYear = Cells(1, i).Characters(5, 4).Text
         Else
            theYear = Cells(1, i).Characters(6, 4).Text
         End If

        ' If theYear > 50 Then 'add prefix to the year
        '    theYear = "19" & theYear
        ' Else
        '    theYear = "20" & theYear
        ' End If

         'check to see if the currentYear is less than first year
         If theYear < firstYear Then
            firstYear = theYear
         End If

         If theYear > lastYear Then
            lastYear = theYear
         End If

         found = False
```

```
    Next i 'done scanning the sheet now redim the array

    'ReDim yearsMonths(firstYear To lastYear, 1 To 12)

End If 'z = 1

ReDim yearsMonths(firstYear To lastYear, 1 To 12)
ReDim contributionEntropy(firstYear To lastYear, 1 To 12)

For i = 4 To numCols 'now process all the nTuple sheets

    'determine current year
    j = 1
    While found = False
        testchar = Sheets(summarySheet).Cells(1, i).Characters(j, 1).Text
        If testchar = "/" Then
            found = True
        Else
        j = j + 1
        'found = True
        End If
    Wend
    'month ends at j

    currentMonth = Sheets(summarySheet).Cells(1, i).Characters(1, j - 1).Text

    If currentMonth < 10 Then
        theYear = Sheets(summarySheet).Cells(1, i).Characters(5, 4).Text
    Else
        theYear = Sheets(summarySheet).Cells(1, i).Characters(6, 4).Text
    End If

    'If theYear > 50 Then 'add prefix to the year
    '    theYear = "19" & theYear
    'Else
    '    theYear = "20" & theYear
    'End If
    '**************

    timeStep = "" & currentMonth & "/" & theYear
    timeStepRow = findStringInSheet(countSheet, timeStep, "A")

  '  temp = Sheets(summarySheet).Cells(1, 1)
  '  Sheets(summarySheet).Cells(1, 1) = timeStepRange
  '  timeStepRow = Sheets(summarySheet).Cells(1, 1).Characters(4, 5).Text
  '  Sheets(summarySheet).Cells(1, 1) = temp

    totalInstances = Sheets(countSheet).Cells(timeStepRow, nTuples + 3)  'O_Q_k
    localSumInstances = Sheets(countSheet).Cells(timeStepRow, z + 2)          'O_q_k

    '
    ' Sh(A)_k_q = num instances A at k in q_level          num instances A at k in q_lvl
    '          -------------------------------                  * log2  --------------------
    '          sum of instances at k in q_lvl                        sum of instances at k in q_lvl

    '
```

```
    ' Sh(k)_q = Sh(A)_k + Sh(B)_k + ...
    '

    '                                    [ sum instances at k in q_lvl (O_q_k)    ]
    'contribution Cs_qlevel_k = abs [ ----------------------------------------   ]     * Sh(k)_q
    '                                    [ sum instances at all Q_lvls (O_Q_k) ]
    '              O_q_k       O_Q_k
    '           +  ----- * log2 -----
    '              O_Q_k       O_q_k
    '

    'traverse all the rows in the summarySheet to get the num of instances of each term
    'and the entropies of each term
    'store the sum of the entropies of each term in step k in the yearsMonths array.

    For j = 2 To numRows
       If Sheets(summarySheet).Cells(j, i) > 0 Then
          theValue = Sheets(summarySheet).Cells(j, i)
          entropy = (-theValue / localSumInstances) * (Log(theValue / localSumInstances) / Log(2))

          yearsMonths(theYear, currentMonth) = yearsMonths(theYear, currentMonth) + entropy
       End If

       'at the last term in the time step compute the contribution entropy
       If j = numRows Then

          contributionEntropy(theYear, currentMonth) = Abs(localSumInstances / totalInstances) _
                        * yearsMonths(theYear, currentMonth) + ((localSumInstances / totalInstances) _
                        * (Log(totalInstances / localSumInstances) / Log(2)))
       End If
    Next j

    found = False
Next i

'output yearsArray

'Sheets(summarySheet).Cells(3, 4) = "Years"
'Sheets(summarySheet).Cells(3, 5) = "instances"

counter = 4 'for output
monthCounter = 1
lastValue = 0
lastContributionValue = 0

For j = firstYear To lastYear

    For k = 1 To 12

       If "" & j = firstYear And k = 1 Then 'label the q level
          Sheets(overallQEntropy).Cells(counter - 1, z + 2) = z
       End If

       If z = 1 Then 'put the month/year
          Sheets(overallQEntropy).Cells(counter, z) = "" & monthCounter & "/" & j
          Sheets(overallQEntropy).Cells(counter, z + 1) = 0
```

```
   Sheets("q_contribution_entropy_monthly").Cells(counter, z) = """ & monthCounter & "/" & j
   Sheets("q_contribution_entropy_monthly").Cells(counter, z + 1) = 0
End If

currentValue = yearsMonths(j, monthCounter)
currentContributionValue = contributionEntropy(j, monthCounter)
If currentValue > 0 Or Not currentValue = Null Then
   Sheets(overallQEntropy).Cells(counter, z + 2) = currentValue

   lastValue = currentValue
Else
   Sheets(overallQEntropy).Cells(counter, z + 2) = lastValue

End If

If currentContributionValue > 0 Or Not currentContributionValue = Null Then
   Sheets(contributionEntropySheet).Cells(counter, z + 2) = currentContributionValue
   lastContributionValue = currentContributionValue
Else
   Sheets(contributionEntropySheet).Cells(counter, z + 2) = lastContributionValue
End If

monthCounter = monthCounter + 1
If monthCounter > 12 Then
   monthCounter = 1
End If

If z = nTuples Then
   Sheets(overallQEntropy).Cells(counter, z + 3) = "=SUM(" & col(2) & counter _
                & ":" & col(nTuples + 2) & counter & ")"
End If

If z = nTuples Then
   'S(q)
   Sheets(contributionEntropySheet).Cells(counter, z + 3) = "=SUM(" & col(2) & counter _
                & ":" & col(nTuples + 2) & counter & ")"

   Sheets(contributionEntropySheet).Cells(counter, z + 5) = _
      Sheets(contributionEntropySheet).Cells(counter, z + 3).Value

   'n(q)
   Sheets(contributionEntropySheet).Cells(counter, z + 4) = _
      Sheets("q_summary_monthly_instances").Cells(counter, z + 3)

   Sheets(contributionEntropySheet).Cells(counter, z + 6) = _
      Sheets(contributionEntropySheet).Cells(counter, z + 4).Value

   If counter > 4 Then 'calc the delta values and temp
      'delta s
      Sheets(contributionEntropySheet).Cells(counter, z + 7) = _
       "=" & col(z + 5) & counter & "-" & col(z + 5) & counter - 1

      'delta n
       Sheets(contributionEntropySheet).Cells(counter, z + 8) = _
       "=" & col(z + 6) & counter & "-" & col(z + 6) & counter - 1
```

```
                'temp delta n / delta s
                Sheets(contributionEntropySheet).Cells(counter, z + 9) = _
                "=" & col(z + 8) & counter & "/" & col(z + 7) & counter
            End If
        End If 'z = nTuples


        counter = counter + 1

    Next k
  Next j

Next z
  'sheet header:
  Sheets(contributionEntropySheet).Cells(2, nTuples + 3) = "S(q)"
  Sheets(contributionEntropySheet).Cells(2, nTuples + 4) = "n(q)"
  Sheets(contributionEntropySheet).Cells(1, nTuples + 5) = "After linear interpolation"
  Sheets(contributionEntropySheet).Cells(2, nTuples + 5) = "S(q)"
  Sheets(contributionEntropySheet).Cells(2, nTuples + 6) = "n(q)"
  Sheets(contributionEntropySheet).Cells(2, nTuples + 7) = "delta S"
  Sheets(contributionEntropySheet).Cells(2, nTuples + 8) = "delta n"
  Sheets(contributionEntropySheet).Cells(2, nTuples + 9) = "temp (n/s)"


'run linear interpolation on S(q) and n(q)
  Call FillInMissingData(contributionEntropySheet, nTuples + 5, 4, counter - 1)
  Call FillInMissingData(contributionEntropySheet, nTuples + 6, 4, counter - 1)

End Sub ' qlevelEntropy




'--------------------------------------------------------------------------
'  sub: qLevelMonthTemp
'  Author: Matt Behnke
'  Created: 2/28/02
'  Description: uses the counts from the q level month sheet to:
'             1) store the instances of each time step into an array
'             2) calculates the probabilities of each instance
'             3) determines the x & y values
'             4) determines the alpha, beta values from the curvefit
'             5) outputs timesteps, and alpha and beta onto a new sheet
'             coeff(1) = A
'             coeff(2) = B
'             alpha = B
'             beta = e^(-A/alpha)
'  inputs:  numqLevels - the number of qlevels nTuples..
'  Outputs:
'--------------------------------------------------------------------------
Sub qLevelMonthTemp(ByVal numQLevels As Integer)

'number of ntuples
nTuples = numQLevels
```

```
Dim instances_q(64) As Variant                    'stores the instances in each q_level q_i_k
Dim probabilities(64) As Variant                  'stores the probabilities of instances P(q_i_k)
Dim x_values(64) As Variant                       'x_values [X: ln(qi+r)]  -weibull
Dim y_values(64) As Variant                       'y_values [Y: ln[-ln(1-P(qi)] -weibull
Dim coefficients As Variant                      'coefficients
Dim numRows As Integer                            'number of rows on the datasheet
Dim numInstances_k As Double                         'total num of instances at step k
Dim gamma As Double                              'the shift, r
Dim numValues As Integer                          'the number of values in the x,y arrays

gamma = 0#

'the source datasheet contains the timesteps and the count of instances in each q level per time step..
datasheet2 = "q_summary_monthly_instances"
tempSheet = "q_level_monthly_temp"

numRows = CountRows(datasheet2, 1)
numValues = nTuples

Sheets.Add After:=Sheets(Sheets.Count)
'Sheets(Sheets.Count).Select
ActiveSheet.Name = tempSheet

Cells(4, 2).Select
ActiveWindow.FreezePanes = True

Sheets(tempSheet).Cells(1, 1) = " "
Sheets(tempSheet).Cells(2, 1) = " "
Sheets(tempSheet).Cells(3, 1) = "k"
Sheets(tempSheet).Cells(3, 2) = "interval"
Sheets(tempSheet).Cells(3, 3) = "A"
Sheets(tempSheet).Cells(3, 4) = "B"
Sheets(tempSheet).Cells(3, 5) = "alpha = B"
Sheets(tempSheet).Cells(3, 6) = "beta = e^(-A/alpha)"
Sheets(tempSheet).Cells(2, 6) = "T = beta"

'numInstances_k = Sheets(datasheet2).Cells(1, 1) 'THe total num of instances over the whole dataset

For k = 4 To numRows 'traverse all the steps
   numInstances_k = Sheets(datasheet2).Cells(k, nTuples + 3) 'columns are offset by 2
   Sheets(tempSheet).Cells(k, 1) = k - 3 'timestep
   Sheets(tempSheet).Cells(k, 2) = Sheets(datasheet2).Cells(k, 1) 'interval

   num_q_at_k = 0

   For z = 1 To nTuples 'traverse all the nTuples
      instances_q(z) = Sheets(datasheet2).Cells(k, z + 2)
      probabilities(z) = 0 'reinit array
      If instances_q(z) > 0 Then
         probabilities(z) = instances_q(z) / numInstances_k
      'ln = log(x) / log(exp(1))
         x_values(z) = Log(instances_q(z) + gamma) / Log(Exp(1)) 'Ln(instances_q(z) + gamma)
         y_values(z) = Log((-Log(1 - probabilities(z)) / Log(Exp(1)))) / Log(Exp(1)) 'Ln(-Ln(1 -
probabilities(z)))
```

```
            num_q_at_k = num_q_at_k + 1

            'DEBUG********8
               Sheets(tempSheet).Cells(3, 10) = "actual probabilities"
               Sheets(tempSheet).Cells(k, z + 9) = probabilities(z)

         Else

         End If
      Next z
      If num_q_at_k > 0 Then
         coefficients = curveFit(num_q_at_k, x_values, y_values, 1)


         Sheets(tempSheet).Cells(k, 3) = coefficients(1) 'A
         Sheets(tempSheet).Cells(k, 4) = coefficients(2) 'B
         Sheets(tempSheet).Cells(k, 5) = coefficients(2) 'alpha = B
         If Not coefficients(2) = 0 Then
            Sheets(tempSheet).Cells(k, 6) = Exp(-coefficients(1) / coefficients(2)) 'beta
         Else
            Sheets(tempSheet).Cells(k, 6) = 0 'beta
         End If
      End If 'q at k > 0
   Next k

   'debug:::::::
   For k = 4 To numRows
      For z = 1 To nTuples

         instances_q(z) = Sheets(datasheet2).Cells(k, z + 2)
         If instances_q(z) > 0 Then
            'DEBUG**************
            '
            ' p(q_i) = 1-e^(q_i/beta)^alpha
            '
            If Sheets(tempSheet).Cells(k, 6) > 0 Then
               Dim p_q_i_calced As Double
               Dim beta As Double
               Dim alpha As Double
               beta = Sheets(tempSheet).Cells(k, 6).Value
               alpha = Sheets(tempSheet).Cells(k, 5).Value
               'Sheets(tempSheet).Cells(1, 1) = instances_q(z)
          '=(1-EXP(B3/$M$7)^$L$7)*-1
            '
               'p_q_i_calced = 1 - (1 / (Exp(instances_q(z) / beta) ^ alpha))
                  Sheets(tempSheet).Cells(3, 40) = "calculated probabilities"
                  Sheets(tempSheet).Cells(k, z + 39) = "=(1-EXP(-" & instances_q(z) & "/" & beta & ")^" &
alpha & ")"
            End If
            '******************
         End If
      Next z
   Next k

   End Sub
```

```
'-------------------------------------------------------------------------
'  function: qLevelGraphStart
'  Author: Matt Behnke
'  Created: 4/26/02
'  Description: Determines the starting position for the source data of the
'               the charts: n(x), S(x), T(x) - Based on Entropy.
'
'               the function checks a column to see when a value is not null and when
'               the value is greater than zero
'
'               only checks the 150 rows..
'
'  inputs: Sheetname - the name of the sheet the function checks.
'          columnNumber - the column number that the function uses
'          rowNumber - starting row
'
'  Outputs: integer
'-------------------------------------------------------------------------
Function qLevelGraphStart(ByVal sheetName As String, ByVal columnNumber As Integer, ByVal
rowNumber As Integer)

    Dim checkValue As String
    For i = rowNumber To rowNumber + 150
        checkValue = Sheets(sheetName).Cells(i, columnNumber)
        If Not checkValue = "" Then
            qLevelGraphStart = i
        Exit For
        End If

    Next i

End Function 'qLevelGraphStart

'-------------------------------------------------------------------------
'  function: qLevelCopyGraphs
'  Author: Matt Behnke
'  Created: 4/26/02
'  Description: Copies the graphs from the AffiliationMacro sheet
'
'  inputs: GraphStart - the starting row of the source data..
'          numQLevels - the number of q levels
'
'  Outputs: none
'-------------------------------------------------------------------------
Sub qLevelCopyGraphs(ByVal graphStart As Integer, ByVal numQLevels As Integer)

    Dim theFilename As String
    Dim macroFilename As String
    Dim copyAfter As String                 'the sheet to copy the graphs after
    Dim lastRow As Integer
    Dim dataOffset As Integer

    dataOffset = numQLevels + 2

    theFilename = Application.ActiveWorkbook.Name
    macroFilename = "AffiliationMacro.xls"
```

```
    copyAfter = "q_level_monthly_temp"
    lastRow = CountRows(copyAfter, 1)

Application.DisplayAlerts = False

 Windows(macroFilename).Activate
    Sheets("T(X) - Weibull Curve Fit").Select
    Sheets("T(X) - Weibull Curve Fit").Copy After:=Workbooks(theFilename).Sheets(copyAfter)
    ActiveChart.PlotArea.Select
    ActiveChart.SeriesCollection(1).Values = "=q_level_monthly_temp!R" & graphStart & "C6:R" &
lastRow & "C6"

    Windows("AffiliationMacro.xls").Activate
    Sheets("S(q)").Select
    Sheets("S(q)").Copy After:=Workbooks(theFilename).Sheets(copyAfter)
    ActiveChart.PlotArea.Select
    ActiveChart.SeriesCollection(1).Values = _
        "=q_contribution_entropy_monthly!R" & graphStart & "C" & dataOffset + 3 & ":R" & lastRow &
"C" & dataOffset + 3

    Windows("AffiliationMacro.xls").Activate
    Sheets("n(q)").Select
    Sheets("n(q)").Copy After:=Workbooks(theFilename).Sheets(copyAfter)
    ActiveChart.PlotArea.Select
    ActiveChart.SeriesCollection(1).Values = _
        "=q_contribution_entropy_monthly!R" & graphStart & "C" & dataOffset + 4 & ":R" & lastRow &
"C" & dataOffset + 4

    Windows("AffiliationMacro.xls").Activate
    Sheets("T(X) - Based on Entropy").Select
    Sheets("T(X) - Based on Entropy").Copy After:=Workbooks(theFilename).Sheets(copyAfter)
    Sheets("T(X) - Based on Entropy").Select
    ActiveChart.PlotArea.Select
    ActiveChart.SeriesCollection(1).Values = _
        "=q_contribution_entropy_monthly!R" & graphStart & "C" & dataOffset + 7 & ":R" & lastRow &
"C" & dataOffset + 7
    ActiveChart.Axes(xlValue).Select


Application.DisplayAlerts = True

End Sub

'--------------------------------------------------------------------------
'  function: qLevelCopyYearsGraph
'  Author: Matt Behnke
'  Created: 4/26/02
'  Description: Copies the graphs from the AffiliationMacro sheet
'
'  inputs: GraphStart - the starting row of the source data..
'          numQLevels - the number of q levels
'
'  Outputs: none
'--------------------------------------------------------------------------
Sub qLevelCopyYearsGraphs(ByVal numQLevels As Integer)
```

```
    Dim theFilename As String
    Dim macroFilename As String
    Dim copyAfter As String                 'the sheet to copy the graphs after
    Dim lastRow As Integer
    Dim dataOffset As Integer

    dataOffset = numQLevels + 2 'last column of data

    theFilename = Application.ActiveWorkbook.Name
    macroFilename = "AffiliationMacro.xls"
    copyAfter = "q_summary_year"
    lastRow = CountRows(copyAfter, 1)

    Application.DisplayAlerts = False

    Windows(macroFilename).Activate
    Sheets("q_level_instances_year").Select
    Sheets("q_level_instances_year").Copy After:=Workbooks(theFilename).Sheets(copyAfter)

    Application.DisplayAlerts = True

    ActiveChart.PlotArea.Select

    'add the series for each year

    ActiveChart.SeriesCollection(1).Name = "=""" & Sheets(copyAfter).Cells(4, 1) & """"
    ActiveChart.SeriesCollection(1).Values = "=" & copyAfter & "!R" & 4 & "C2:R" & 4 & "C" &
dataOffset
    ActiveChart.SeriesCollection(1).Select
    With Selection
        .Smooth = True
    End With


    For i = 5 To lastRow 'offset is 3.. 5 - 3 = 2
        ActiveChart.SeriesCollection.NewSeries
        ActiveChart.SeriesCollection(i - 3).Name = "=""" & Sheets(copyAfter).Cells(i, 1) & """"
        ActiveChart.SeriesCollection(i - 3).Values = "=" & copyAfter & "!R" & i & "C2:R" & i & "C" &
dataOffset
        ActiveChart.SeriesCollection(i - 3).Select
        With Selection
            .Smooth = True
        End With
    Next i

End Sub 'copy years chart
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX Q – LIST OF TECHNOLOGY TRANSFER MODELS

Below is a screen shot from Dr. Saboe's dissertation:

indication that the model proposed in this research addresses that feature. Each of these models in Table 1 are summarized in the following sections.

Model In Tech Tx Literature	Model Feature	Proposed Information/Control Theory Model Contribution
Theory of Human Needs Model	Complexity factor framework facts, perceptions, actions	Learning Curve Actions on messages (tasks)
Structure Changes Model–	Internal and External Relationship	Shannon Entropy of Messages Joint entropy Information In, Information Out
Technology Model	Goodness of Technology Alone causes Diffusion	Identifies Minimum number of nodes (senders and receptors) extensions may address vacuum and pressure
Institution Building Model	External Influences affect the human behavior to assimilate a technology	Identifies Entropy as a factor that can influence the acceptance of a technology
Equilibrium vs Conflict Model	Equilibrium is an Instrument for Balance Conflict Is a Instrument to apply Pressure	Convergence in Entropy yields balance Large Differences in Entropy yields Pressure
Communication Model	A Technology is Delivered to Adopters Through a Channel, If Understood It is acted upon.	Quantifies the Information Being Acted on, and Quantifies notion of "Understood " in terms of Entropy and Learning Curve
Problem solving Model	Present Hypothesis Test Hypothesis with Data and Logic	Hypothesizes a Mathematical Model and Explains based on Data Analysis

Table II-1 Technology Transfer Models, Features, and Relation to Proposed Model

- 31 -

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

Defense Technical Information Center
Ft. Belvoir, VA

Dudley Knox Library
Naval Postgraduate School
Monterey, CA

National Technical Information Service
Springfield VA

Dr. Luqi
Naval Postgraduate School
Monterey, CA

Dr. Mantak Shing
Naval Postgraduate School
Monterey, CA

U.S. Army Tank-automotive & Armaments Command,
Tank Automotive Research, Development and Engineering Center
Associate Director
Next Generation Software Engineering
ATTN: AMSTA-TR-R/265
Warren MI

Christopher D. Miles
U.S. Army Next Generation Software Engineering (AMSTA-TR-R/265)
U.S. Tank-Automotive & Ammunition Command
Warren, MI

Nimitz Library
U.S. Naval Academy
Annapolis, MD

Laura Malone
Software Engineering Institute
Software Engineering Information Repository
Carnegie Mellon University
Pittsburgh, PA

Barker Engineering Library
Massachusetts Institute of Technology
Cambridge, MA

Library of Congress
Copyright Office
Washington, D.C.

Matthew J. Behnke
U.S. Army Next Generation Software Engineering (AMSTA-TR-R/265)
U.S. Tank-Automotive & Ammunition Command
Warren, MI